Towards End-to-End resource provisioning in Fog Computing over Low Power Wide Area Networks

José Santos[a,*], Tim Wauters[a], Bruno Volckaert[a] and Filip De Turck[a]

[a]Ghent University - imec, IDLab, Department of Information Technology, Technologiepark-Zwijnaarde 126, 9052 Gent, Belgium

ARTICLE INFO

Keywords:
Resource Provisioning
Fog Computing
IoT
Service Function Chaining
MILP

ABSTRACT

Recently, with the advent of the Internet of Things (IoT), Smart Cities have emerged as a potential business opportunity for most cloud service providers. However, centralized cloud architectures cannot sustain the requirements imposed by many IoT services. High mobility coverage and low latency constraints are among the strictest requirements, making centralized solutions impractical. In response, theoretical foundations of Fog Computing have been introduced to set up a distributed cloud infrastructure by placing computational resources close to end-users. However, the acceptance of its foundational concepts is still in its early stages. A key challenge still to answer is Service Function Chaining (SFC) in Fog Computing, in which services are connected in a specific order forming a service chain to fully leverage on network softwarization. Also, Low Power Wide Area Networks (LPWANs) have been getting significant attention. Opposed to traditional wireless technologies, LPWANs are focused on low bandwidth communications over long ranges. Despite their tremendous potential, many challenges still arise concerning the deployment and management of these technologies, making their wide adoption difficult for most service providers. In this article, a Mixed Integer Linear Programming (MILP) formulation for the IoT service allocation problem is proposed, which takes SFC concepts, different LPWAN technologies and multiple optimization objectives into account. To the best of our knowledge, our work goes beyond the current state-of-the-art by providing a complete end-to-end (E2E) resource provisioning in Fog-cloud environments while considering cloud and wireless network requirements. Evaluations have been performed to evaluate in detail the proposed MILP formulation for Smart City use cases. Results show clear trade-offs between the different provisioning strategies. Our work can serve as a benchmark for resource provisioning research in Fog-cloud environments since the model approach is generic and can be applied to a wide range of IoT use cases.

1. Introduction

In recent years, the Internet of Things (IoT) has introduced a whole new set of challenges and opportunities by converting everyday life objects into smart communicating devices [1]. Due to the advent of IoT and the wide adoption of virtualization and cloud technologies, the concept of Smart Cities has become an even more attractive business opportunity [2]. Smart Cities powered by IoT aim to revolutionize different domains of urban life. For instance, improving public transportation and environmental monitoring. According to Cisco [3], billions of devices will be integrated in the IoT ecosystem in the forthcoming years. All these devices will be connected to the network, sending and receiving data to the cloud, making current centralized cloud solutions impractical. As an answer, Fog Computing [4, 5] has emerged as an extension to the cloud paradigm, by bringing cloud services closer to the end devices, thus, helping to meet the demanding requirements introduced by IoT services (e.g. low latency, high mobility coverage). Waste management platforms and surveillance camera systems are already envisioned Smart City use cases for Fog-cloud infrastructures, which will benefit from the nearby real-time processing, storage procedures and data analytics to overcome the limitations of centralized cloud environments [6].

Furthermore, IoT is pushing for a paradigm shift in terms of connectivity for these so-called smart devices. Recently, Low Power Wide Area Networks (LPWANs) have drawn significant attention [7]. These wireless solutions enable low bandwidth communications over long ranges, up to several kilometers, at low power consumption, ensuring a high device lifetime. LoRaWAN [8], Sigfox [9] and the upcoming IEEE 802.11ah standard [10] are among the most popular LPWAN technologies today. In spite of their significant potential to impact the IoT ecosystem, the current market is highly fragmented and many challenges still exist concerning the deployment and management of these technologies, making their wide adoption difficult for most service providers. Additionally, micro-services are currently revolutionizing software development practices [11]. An application is decomposed in a set of small, self-contained containers deployed across a large number of servers, as opposed to the traditional single monolithic application. Containers are currently the most promising alternative to the conventional Virtual Machines (VMs), due to their high scalability and low overhead. Nevertheless, several challenges still prevent cloud providers and end users from fully benefiting from network virtualization and micro-service patterns. For example, users access a database to retrieve collected data, in which data values were already filtered and modified by a machine learning engine. Cloud providers must implement proper allocation strategies to ensure that database services are instantiated close to the users and that machine learning services are allocated nearby database ones to reduce the latency in accessing the inferred results. This key challenge is

josepedro.pereiradossantos@UGent.be (J. Santos)
ORCID(s): 0000-0002-6276-2057 (J. Santos); 0000-0003-2618-3311 (T. Wauters); 0000-0003-0575-5894 (B. Volckaert); 0000-0003-4824-1199 (F.D. Turck)

called Service Function Chaining (SFC) [12, 13], where services must be connected in a specific order, forming a service chain that each request needs to traverse to access a particular Network Service (NS). Thus, NSs are dynamically configured in software without any significant change at the hardware level, which results in optimized network resources and increased application performance. Although SFC provides high flexibility and low operational costs, SFC concepts are still quite unexplored in Fog Computing since most SFC research is focused on Multi-access Edge Computing (MEC) use cases, in which the interactions between fog locations and the cloud are not considered [14]. In MEC scenarios, all services are preferably allocated at the network edge close to end-users to reduce latency and avoid congestion in the network core while in Fog Computing, services can be placed in a fog node or cloudlet [15], but also in the cloud making the inherent bi-directional communications crucial due to the hierarchical architecture. For example, a service may be allocated in the cloud due to its high computational requirements but needs to interact with another service, which may be located in a Fog location. These interactions (e.g. low latency, minimum available bandwidth) must be guaranteed. These bi-directional communications are currently not being studied in the context of MEC.

Although the theoretical foundations of Fog Computing have been already established, its adoption is in early stages. Research challenges in terms of resource provisioning and service scheduling persist. Therefore, in this article, a Mixed-Integer Linear Programming (MILP) formulation for the IoT service allocation problem is proposed, which takes SFC concepts, different LPWAN technologies and multiple optimization objectives into consideration. To the best of our knowledge, our work goes beyond the current state-of-the-art by bridging the gap between the cloud and the wireless domain, since most research only focuses on one of the domains and almost no consideration is given to the joint optimization of both. Finally, evaluations have been performed to validate our proposal, specifically for Smart City use cases. The proposed MILP model optimizes SFC allocation in Fog-cloud environments by not only reducing user latency since services (e.g. databases) are instantiated close to users, but also by decreasing the sensor's data transfer time and taking bandwidth requirements into account. The result of our work can serve as a benchmark in research covering IoT provisioning issues in Fog Computing since the model approach is generic, considers several cloud and wireless aspects and can be applied to a wide range of IoT use cases. Furthermore, our model can be adopted in realistic scenarios as the ones presented in Section 4 and the performance of future heuristics can be assessed based on our measured results.

The remainder of the article is organized as follows. In the next Section, related work is discussed. Section III introduces the proposed MILP model for the IoT service allocation problem. In Section IV, the use cases are described which is followed by the evaluation setup in Section V. Next, in Section VI, results are shown. Finally, conclusions are presented in Section VII.

2. Related Work

Resource provisioning or also known as resource allocation has been studied for years in the network management domain [16], [17]. In recent years, resource provisioning and service placement issues gained significant attention in the field of Fog Computing. In [18], an optimization formulation for the service deployment of IoT applications in Fog scenarios has been proposed and implemented as a prototype called FogTorch. Their approach focused not only on hardware and software demands, but also on Quality of Service (QoS) requirements, such as network latency and bandwidth. Results showed that their heuristic algorithm ensures the optimal deployment of services while decreasing hardware capacity and increasing resource demands. Additionally, in [19], the IoT service placement issue has been modeled as an optimization problem. The model focused on the maximization of Fog resources and the minimization of overall network delay. Their work has been extended in [20], where application QoS metrics and deadlines for the provisioning of each type of application have been taken into account in their ILP formulation. In [21] both architectural and resource allocation concepts have been tackled. The authors presented a provisioning algorithm focused on service elasticity and on the optimization of available resources. Simulation results showed that the proposed algorithm efficiently allocates resources while minimizing the response time and maximizing the throughput. Furthermore, in [22], a particle swarm optimization algorithm has been introduced for the resource provisioning in Fog-cloud infrastructures specifically focused on Smart buildings. Results showed that their approach can reduce the response time, the data transfer and the cost of VM allocation. In [23], a provisioning algorithm formulated as a Mixed-Integer Non-Linear (MINL) problem has been presented for VM allocation in IoT networks. Their proposal enables the offloading of computationally intensive and delay-sensitive tasks to Fog nodes connected to IoT gateways. Their goal is to minimize the system cost by meeting QoS requirements. Then, in [24], their work has been extended by evaluating the trade-off between maximizing the reliability and minimizing the overall system cost. A highly computationally complex ILP model has been described followed by a heuristic-based algorithm able to find suboptimal solutions, albeit achieving better time efficiency. Additionally, in [25], a MILP formulation addressing the MEC resource provisioning issue for IoT services has been introduced. Their approach focused on the provisioning of resources (edge servers and applications) as well as the workload assignment while minimizing latency. Moreover, in [26], two placement strategies in Fog Computing based on matching game algorithms have been introduced. The first one is based on SFC concepts, by taking into account the ordered sequence of services requested by each application. The second one overlooks the SFC structure, to lower the computation complexity without losing performance. Comparison results highlighted the increased performance of the stated methods. In [27], the authors proposed a Fog Computing scheme to support IoT-based crowdsensing applications. Their proposal focused on a MINL formulation, which has then been linearized into a MILP model. Results showed that their proposal can outperform traditional cloud infrastructures. Additionally, in [28], a scheduling mechanism for resource provisioning in Fog Computing based on deep reinforcement learning has been presented. The work focuses on minimizing the time consumption of safety-related applications in vehicular fog use cases. Results confirmed that their allocation schemes can reduce time consumption when compared to traditional cloud infrastructures. Recently, in [29], a provisioning approach for Fog Computing based on Bayesian learning techniques has been presented. Their work focuses on the dynamic scaling of resources according to the current network demand. Simulation results have shown that their proposal can reduce costs and delay violations. Furthermore, in [30], a resource allocation approach focused on dynamic deadline-based requirements has been proposed. Their proposal provisions resources in a hierarchical manner by considering dynamic changes in user requirements and the limited available resources in Fog nodes. Simulation results have shown improvements in terms of data processing time, allocation costs and network delay when compared with other approaches.

Although most of the cited research has dealt with provisioning issues in Fog Computing, none of the aforementioned studies considered realistic latency-sensitive services with actual E2E latency demands, or any kind of connectivity constraints coming from the wireless domain. Also, few works considered the strict requirements coming from service chains or container-based applications. Most studies are focused on VM allocations, while container-based provisioning is still a novel research topic. Thus, in this article, a MILP formulation has been proposed to tackle the problem of resource provisioning in Fog-cloud environments focused on containerized services. The present work builds further on [31] by considering SFC concepts, different LPWAN technologies and several optimization objectives. By combining characteristics coming from the cloud and the wireless domain, our approach paves the way towards a complete E2E resource provisioning in the Smart City ecosystem.

3. The MILP Model: E2E Service Provisioning in Fog-cloud Infrastructures

This section introduces the MILP formulation for the IoT service allocation problem. Then, the variables considered in the model are described, followed by the objectives and respective constraints.

3.1. Model Description

The proposed MILP model significantly extends the authors' recent previous work [31] as follows: the previous formulation has already considered cloud and wireless characteristics. Nevertheless, several additions have been made to the model mainly dealing with service chaining and LPWAN characteristics. The most relevant ones are the limited bandwidth capacity of cloud and Fog nodes and the minimum bandwidth requirement of micro-services. Then, latency metrics have been included instead of hop count. Several SFC and Network Slicing concepts have been added. LoRaWAN has been also included as an LPWAN technology. Then, micro-service replication and the gateway bandwidth factor have been added as a decision variable. Finally, the sensors' data transfer time and the user latency have been included as an optimization objective.

The model decomposes an IoT application in a set of different micro-services, which have a replication factor. Multiple users are expected to access these micro-services. Sensors are spread across the network area to collect data. Each sensor needs to be connected with a wireless gateway to be able to send data to the Fog-cloud infrastructure. The bandwidth available per sensor is affected by a gateway bandwidth factor, which may increase the sensor's data transfer time. The Fog-cloud infrastructure manages a set of nodes, in which micro-service instances must be allocated based on its requirements and subject to multiple constraints:

- Nodes have limited capacities (e.g. CPU and memory).
- Micro-service instances cannot be instantiated on every node, due to specific hardware or software requirements.
- Gateways have limited capacity, based on a maximum number of association identifiers (AIDs).
- Sensors can only associate with gateways within their communication range.
- Sensors need to be associated with the gateway slice allocated for the particular application which they are trying to access.
- The sensors' data transfer time depends on the selected LPWAN technology.
- All micro-services composing an application must be allocated in the network.
- Users must access their assigned application.
- The replication factor of each micro-service depends on the number of user requests.

The model incorporates multiple optimization objectives, which are executed iteratively. In each iteration, a different optimization objective is considered. Additional constraints are added to the model to retain the objective values obtained in previous iterations, imposing an upper or lower bound. Consequently, since iterations must satisfy the previous optimal solutions, the solution space continuously decreases. Every iteration refines the previously obtained solution by improving the model with an additional optimization objective. Sequential objectives have been preferred to multi-criteria objectives to reduce the complexity of the MILP model calculation. Furthermore, opposing allocation policies have been applied in the evaluation, thus, the sequential ordering of objectives eased the shift between strategies and refining the solution space. The objectives considered in the model are the following:

1) Maximization of accepted user requests.
2) Minimization of service migrations between iterations.

Table 1
Input variables related to the cloud infrastructure

Symbol	Description
N	The set of nodes on which micro-service instances are executed.
A	The set of all IoT applications. Each application is composed of a set of different micro-services.
S	The set of all micro-services.
$\underline{ID}$	The set of SFC identifiers.
$\underline{U}$	The set of users.
L	The set of locations where users can access a given application $a \, \varepsilon \, A$.
$\underline{\Phi_{u,a}}$	The user assignment matrix. If $\Phi_{u,a} = 1$, the user u makes use of the application a.
$\underline{\rho_a}$	The maximum number of users that can associate with an application.
$\underline{\rho_s}$	The maximum number of users that can associate with a micro-service instance.
$\underline{\lambda_u}$	The association cost of the user u for the assigned application.
$\underline{\beta}$	The maximum replication factor for each micro-service.
$\underline{\upsilon_s}$	The SFC first position indicator. If $\upsilon_s = 1$, the micro-service s is the first micro-service in the service chain.
$\underline{\iota_s}$	The SFC last position indicator. If $\iota_s = 1$, the micro-service s is the last micro-service in the service chain.
$\underline{\mu}$	The service migration factor represents the maximum allowed percentage of micro-service reallocations.
$I_{a,s}$	The Instance matrix. If $I_{a,s} = 1$, the micro-service s is part of application a. Otherwise, the micro-service s is not part of application a.
$R_{s,n}$	The Relation matrix. If $R_{s,n} = 1$, the micro-service s can be allocated on node n. Otherwise, it cannot be instantiated on node n due to hardware or software limitations.
Ω_n	The total CPU capacity (in cpu) of the node $n \, \varepsilon \, N$.
Γ_n	The total memory capacity (in GB) of the node $n \, \varepsilon \, N$.
$\underline{\Delta_n}$	The bandwidth capacity (in Mbit/s) of the node $n \, \varepsilon \, N$.
ω_s	The CPU requirement (in cpu) of the micro-service $s \, \varepsilon \, S$.
γ_s	The memory requirement (in GB) of the micro-service $s \, \varepsilon \, S$.
$\underline{\delta_s}$	The bandwidth requirement (in Mbit/s) of the micro-service $s \, \varepsilon \, S$.
B_{n_1,n_2}	The bandwidth matrix indicates the available bandwidth capacity (Mbit/s) between the node n_1 and the node n_2.
$\underline{\tau_{n_1,n_2}}$	The latency matrix indicates the latency (in ms) between the node n_1 and the node n_2.
$\underline{\tau_{l_1,l_2}}$	The location matrix indicates the latency (in ms) between the location l_1 and the location l_2.
$\underline{\tau_{n,l}}$	The node location matrix indicates the latency (in ms) between node n and location l.
$\underline{\tau_{u,l}}$	The user location matrix indicates the latency (in ms) between user u and location l.
$\underline{\tau_{u,n}}$	The user node matrix indicates the latency (in ms) between user u and node n.
C_{s_i,s_j}	The Communication matrix indicates the minimum amount of bandwidth (in Mbit/s) between the micro-services s_i and s_j for their proper operation. The micro-service s_i is the source of the network flow while s_j is the sink.
$E_{n,l}$	If $E_{n,l} = 1$, the node n is at location l.
$\underline{E_{u,l}}$	If $E_{u,l} = 1$, the user u is at location l.
$\underline{\alpha_{s_i,s_j}}$	The service matrix. If $\alpha_{s_i,s_j} = 1$, the flow bandwidth between the micro-service s_i and the micro-service s_j needs to be guaranteed. If $\alpha_{s_i,s_j} = 0$, the flow bandwidth does not need to be guaranteed.

3) Minimization of the number of active nodes.
4) Minimization of the number of active gateways.
5) Minimization of the network latency.
6) Minimization of the sensor's data transfer time.
7) Minimization of the user latency.

3.2. Variables

The input variables used in the model are shown in Table 1 and in Table 2, while decision variables are shown in Table 3. All variables added to previous work have been underlined. Thirty-one new input variables have been included in the model. Furthermore, thirteen new decision variables have been added to the model, while two others have been slightly modified to address micro-service replication. Regarding cloud formulation, a set of applications A composed of micro-services S need to be allocated on nodes $n \, \varepsilon \, N$. Each application a has a given SFC identifier $id \, \varepsilon \, ID$. All micro-services have a maximum number of replicas given by β. The replication factor for a particular micro-service $s \, \varepsilon \, S$ for the application a with the SFC identifier id is given by $\beta_{a,id,s}$. Thus, the model determines the exact number of replicas for each micro-service depending on the considered objective (e.g. maximizing user requests, reducing user latency). Each micro-service s has a CPU and a memory requirement represented by ω_s (in cpu) and γ_s (in GB) respectively. For instance, a CPU requirement equal to 0.5 cpu (i.e. 500 millicpu) means that the micro-service needs 50% of a core to operate properly. Also, each micro-service s has a minimum bandwidth requirement represented by δ_s (in Mbit/s). A binary placement matrix P is used to represent

Table 2
Input variables related to the wireless dimensioning

Symbol	Description
GW	The set of wireless gateways.
SR	The set of sensors.
SL	The set of network slices.
Θ_{gw}	The total association identifiers (AIDs) available on a gateway $gw \in GW$.
θ_{sr}	Each sensor $sr \in SR$ needs an AID to connect with a gateway.
$\underline{\Phi_{sr,a}}$	The sensor assignment matrix. If $\Phi_{sr,a} = 1$, the sensor sr makes use of the application a.
$\underline{D_{gw,sr}}$	The Distance matrix indicates the distance (in meters) between the gateway gw and the sensor sr.
$PL_{gw,sr}$	The Path Loss matrix indicates the path loss (in dB) between the gateway gw and the sensor sr.
$A_{sr,gw}$	The Association matrix. If $A_{sr,gw} = 1$, the sensor sr can associate with the gateway gw.
$\underline{A_{a,sl}}$	The Application Slice matrix. If $A_{a,sl} = 1$, the slice sl is responsible for the wireless traffic belonging to app. a.
$\underline{A_{sr,sl}}$	The Slice Association matrix. If $A_{sr,sl} = 1$, the sensor sr needs to be assigned to the slice sl due to its associated application.
$\underline{B_{sl,gw}}$	The Bandwidth matrix indicates the available bandwidth capacity (in Mbit/s) for the slice sl in the gateway gw.
$\underline{\eta_{sr,sl,gw}}$	The Data Rate matrix indicates the available bandwidth (in Mbit/s) for the sensor sr assigned to slice sl in the gateway gw.
$\underline{\tau_{gw,l}}$	The gateway location matrix indicates the latency (in ms) between gateway gw and location l.
$\underline{\tau_{gw,n}}$	The gateway node matrix indicates the latency (in ms) between gateway gw and node n.
$\underline{I_{gw}}$	The IEEE 802.11 ah matrix. If $I_{gw} = 1$, the gateway gw is an IEEE 802.11 ah gateway.
$\underline{L_{gw}}$	The LoRaWAN matrix. If $L_{gw} = 1$, the gateway gw is a LoRaWAN gateway.
$\underline{K_{gw}}$	The gateway access delay matrix represents the propagation time (in ms) from the gateway gw to the Fog-cloud infrastructure.
$\underline{\pi_a}$	The number of bits needed to be transmitted in each upload message for application a.
$\underline{\pi_{sr}}$	The number of bits needed to be transmitted by each sensor sr based on the assigned application.
$E_{gw,l}$	If $E_{gw,l} = 1$, the gateway gw is at location l.
$E_{sr,l}$	If $E_{sr,l} = 1$, the sensor sr is at location l.

in which node n, the replica β_i of a micro-service s is allocated. Also, the previous placement matrix $P^{a,id}_{s,\beta_i}(n)^{i-1}$ is added to the model so that decisions can be made in terms of service migrations since the model has information on where micro-service instances have been allocated on the previous iteration. Then, the service migrations matrix M is used to indicate if a particular micro-service replica has been reallocated on another node. If $M^{a,id}_{s,\beta_i} = 1$, the replica β_i of micro-service s for the application a with the SFC identifier id has been reallocated based on the previous placement matrix. Additionally, the service migration factor μ represents the maximum allowed percentage of service reallocations between model iterations.

Regarding wireless formulation, the model supports two LPWAN technologies, which have been represented through linear equations: IEEE 802.11ah and LoRaWAN. The LoRaWAN formulations are based on the work presented in [32]. The binary matrix I_{gw} indicates if the gateway gw is an IEEE 802.11ah gateway, while the binary matrix L_{gw} determines if the gateway gw is a LoRaWAN gateway. Also, the gateway access delay matrix given by K_{gw} represents the access delay from the gateway gw to the Fog-cloud infrastructure. Without loss of generality, if $I_{gw} = 1$ (i.e. IEEE 802.11 ah gateway), the access delay corresponds to 2 ms. Otherwise, if $L_{gw} = 1$ (i.e. LoRaWAN gateway), the access delay is equal to 5 ms. The distance matrix D indicates the distance (in meters) between a gateway gw and a sensor sr. An additional binary association matrix A is used to indicate if a sensor sr can associate with a gateway gw. This association is based on the distance matrix D and on the AIDs available on each gateway. An IEEE 802.11ah [33] gateway cannot have more than 8192 associated stations as stated in the latest standard. However, the association limitation has been set to 50, since an urban macro deployment with extended range has been considered [34]. For LoRaWAN gateways, the association limitation has been set to 100, since the maximum data rate achieved by each sensor decreases considerably when a higher number of sensors is associated [35]. Thus, by setting up a lower limitation, it is assumed that good channel conditions are always achieved and that all sensors can access the deployed applications, if connected with one gateway. Furthermore, for IEEE 802.11 ah, the distance limitation is set to one thousand meters because this is the maximum coverage range in IEEE 802.11ah networks [36] while for LoRaWAN the limit is set to four thousand meters [37]. If $D_{gw,sr}$ is lower than the imposed limit, the sensor sr can associate with gateway gw and then $A_{sr,gw} = 1$, otherwise, $A_{sr,gw} = 0$. Furthermore, the concept of network slicing has been included in the model by adding the set of slices SL. The binary application slice matrix $A_{a,sl}$ indicates if the slice sl is responsible for the wireless traffic belonging to application a while the binary slice association

Table 3
Decision variables of the MILP model

Symbol	Description
$\underline{G_{a,id}}$	The application acceptance matrix. If $G_{a,id} = 1$, the application a with the SFC identifier id can be allocated. If $G_{a,id} = 0$, the application a with the SFC identifier id cannot be deployed.
$\underline{G_{a,id,s}}$	The micro-service acceptance matrix. If $G_{a,id,s} = 1$, the micro-service s for the application a with the SFC identifier id can be allocated. If $G_{a,id,s} = 0$, the micro-service s for the application a with the SFC identifier id cannot be deployed.
$\underline{G_{u,a,id}}$	The user acceptance matrix. If $G_{u,a,id} = 1$, the user u is associated with application a with the SFC identifier id.
$\underline{\beta_{a,id,s}}$	The replication factor of the micro-service s for the application a with the SFC identifier id.
$\underline{P^{a,id}_{s,\beta_i}(n)}$	The placement matrix. If $P^{a,id}_{s,\beta_i}(n) = 1$, the replica β_i of micro-service s is executed on node n for the application a with the SFC identifier id.
$\underline{P^{a,id}_{s,\beta_i}(n)^{i-1}}$	The placement matrix from the previous iteration.
$\underline{F^{a,id,s_i,\beta_i}_{s_j,\beta_j}(n_1,n_2)}$	The binary flow matrix indicates that the replica β_i from the micro-service s_i and the replica β_j from the micro-service s_j are allocated on node n_1 and n_2, respectively, for the application a with the SFC identifier id.
$\underline{\Lambda_{a,id}}$	The SFC latency matrix indicates the time (in ms) to traverse all possible paths in the service chain for the application a with the SFC identifier id.
$\underline{M^{a,id}_{s,\beta_i}}$	The service migrations matrix. If $M^{a,id}_{s,\beta_i} = 1$, the replica β_i of micro-service s for the application a with the SFC identifier id has been reallocated to another node based on the previous iteration.
U_{gw}	The gateway utilization matrix. $U_{gw} = 1$ indicates that there is at least one sensor associated with gateway gw.
U_n	The node utilization matrix. $U_n = 1$ indicates that there is at least one micro-service replica allocated on node n.
$U_{s,n}$	The micro-service execution matrix. If $U_{s,n} = 1$, a replica of the micro-service s is allocated on node n. If $U_{s,n} = 0$, no replica of the micro-service s is allocated on node n.
$\underline{U^{u,a,id}_{s,\beta_i}(n)}$	The user service matrix. If $U^{u,a}_{id,s,\beta_i}(n) = 1$, the user u is associated with the replica β_i of micro-service s allocated on node n for the application a with the SFC identifier id.
$\underline{U_{sr,gw}}$	The sensor association matrix. If $U_{sr,gw} = 1$, the sensor sr is associated with gateway gw.
$\underline{U_{sr,sl}}$	The slice association matrix. If $U_{sr,sl} = 1$, the sensor sr is associated with the slice sl.
$\underline{U_{sr,sl,gw}}$	The sensor execution matrix. If $U_{sr,sl,gw} = 1$, the sensor sr is assigned to slice sl in the gateway gw.
$\underline{\zeta_{gw}}$	The gateway bandwidth factor.
$\underline{T_{sr}}$	The data transfer time matrix indicates the propagation time (in ms) of a single upload message from the sensor sr.
$\underline{\psi_u}$	The user latency matrix indicates the propagation time (in ms) of a request from the user u to reach the last micro-service in the assigned service chain.

matrix $A_{sr,sl}$ indicates if the sensor sr needs to be associated with slice sl due to its assigned application. The bandwidth matrix $B_{sl,gw}$ contains the available bandwidth (in Mbit/s) for the slice sl in the gateway gw. Then, the data rate matrix $\eta_{sr,sl,gw}$ contains the available bandwidth for the sensor sr assigned to slice sl in the gateway gw. For LoRaWAN, the data rate of each gw depends on the Spreading Factor (SF) adopted by each sensor sr. The SF concerns the ratio between the symbol rate and the chip rate. LoRaWAN spreads each symbol in a rate of 2^{SF} chips per symbol with $SF = \{7, .., 12\}$. Increasing the spreading factor reduces the transmitted data rate at the expense of offering longer range. In the model, SF has been set to 9. Additionally, the data transfer time matrix T_{sr} corresponds to the propagation time (in ms) for a single upload message from the sensor sr to reach the associated gateway. The data transfer time depends on the selected LPWAN technology and on the total number of connected sensors to the particular gateway, which is affected by ζ_{gw}, the gateway bandwidth factor. Essentially, the available bandwidth per sensor may drop depending on the number of connected sensors on each gateway. Further details on how data transfer time is affected by this factor are given in the next section. Then, to fully minimize the latency expected by each user when accessing the given application, another decision variable ψ_u has been added, which contains the propagation time (in ms) of a request from the user u to reach the assigned application, more precisely to access an instance of the last micro-service in the application's service chain with which user u is connected to. Each optimization objective is detailed below. All constraints previously presented in [31] have been considered in this extended model. To avoid repetition, only constraints related to novel variables are described.

3.3. Optimization Objectives & Constraints

3.3.1. *Maximization of User Requests (MAX R)*

This objective is related to the maximization of acceptance of user requests. Multiple constraints have been added to reflect the extensions regarding service replication, service chaining and wireless formulations made in the model. Firstly, the objective has been updated to consider the user's assigned application as shown in (1).

$$max \sum_{u \,\varepsilon\, U} \sum_{a \,\varepsilon\, A} \sum_{id \,\varepsilon\, ID} \Phi_{u,a} \times G_{u,a,r} \tag{1}$$

To limit the association of users to a certain application, a constraint represented by (2) has been used to guarantee that the maximum number of users per application is respected.

$$\forall a \,\varepsilon\, A, id \,\varepsilon\, ID : \sum_{u \,\varepsilon\, U} \Phi_{u,a} \times G_{u,a,id} \le \rho_a \tag{2}$$

A constraint has been also included to ensure that each user is associated with only one application as shown in (3).

$$\forall u \,\varepsilon\, U : \sum_{a \,\varepsilon\, A} \sum_{id \,\varepsilon\, ID} \Phi_{u,a} \times G_{u,a,id} = 1 \tag{3}$$

Service association constraints have also been added to guarantee that users are associated with one particular micro-service instance for all micro-services of their admitted application. Thus, users can only access their assigned application if they are associated with all of their micro-services as shown in (4). Also, each user can only be admitted to only one replica of the same micro-service as given by (5).

$$\begin{aligned}&\forall u \,\varepsilon\, U, a \,\varepsilon\, A, id \,\varepsilon\, ID : \\ &\sum_{s \,\varepsilon\, S} \sum_{\beta_i \,\varepsilon\, \beta} \sum_{n \,\varepsilon\, N} U^{u,a,id}_{s,\beta_i}(n) = \Phi_{u,a} \times \sum_{s \,\varepsilon\, S} I_{a,s}\end{aligned} \tag{4}$$

$$\begin{aligned}&\forall u \,\varepsilon\, U, a \,\varepsilon\, A, id \,\varepsilon\, ID, s \,\varepsilon\, S : \\ &\sum_{\beta_i \,\varepsilon\, \beta} \sum_{n \,\varepsilon\, N} U^{u,a,id}_{s,\beta_i}(n) = \Phi_{u,a} \times I_{a,s}\end{aligned} \tag{5}$$

Each micro-service replica is then subject to an association limitation based on ρ_s and the user's association cost λ_u as shown in (6).

$$\begin{aligned}&\forall a \,\varepsilon\, A, id \,\varepsilon\, ID, s \,\varepsilon\, S, \beta_i \,\varepsilon\, \beta, n \,\varepsilon\, N : \\ &\sum_{u \,\varepsilon\, U} \lambda_u \times \Phi_{u,a} \times U^{u,a,id}_{s,\beta_i}(n) \le \rho_s\end{aligned} \tag{6}$$

Also, a constraint has been included to assure that users are only connected with micro-service instances allocated in the network as shown in (7).

$$\begin{aligned}&\forall u \,\varepsilon\, U, \forall a \,\varepsilon\, A, id \,\varepsilon\, ID, s \,\varepsilon\, S, \beta_i \,\varepsilon\, \beta, n \,\varepsilon\, N : \\ &U^{u,a,id}_{s,\beta_i}(n) \le P^{a,id}_{s,\beta_i}(n)\end{aligned} \tag{7}$$

Then, two constraints have been added to reflect the relations between the acceptance matrices $\boldsymbol{G}$. A micro-service is considered instantiated if the expected number of micro-service instances is equal to the number of allocated replicas in the network. Also, an application is only accepted if all its micro-services have been admitted in the network (i.e. all instances deployed on the network). These constraints are represented by (8) and (9), respectively.

$$\begin{aligned}&\forall a \,\varepsilon\, A, id \,\varepsilon\, ID, s \,\varepsilon\, S : \\ &G_{a,id,s} = \begin{cases} 1.0 & \text{if } I_{a,s} \times \beta_{a,id,s} = \sum_{\beta_i, n \,\varepsilon\, \beta, N} P^{a,id}_{s,\beta_i}(n) \\ 0.0 & \text{Otherwise} \end{cases}\end{aligned} \tag{8}$$

$$\begin{aligned}&\forall a \,\varepsilon\, A, id \,\varepsilon\, ID : \\ &G_{a,id} = \begin{cases} 1.0 & \text{if } \sum_{s \,\varepsilon\, S} I_{a,s} = \sum_{s \,\varepsilon\, S} G_{a,id,s} \\ 0.0 & \text{Otherwise} \end{cases}\end{aligned} \tag{9}$$

A constraint has been added to limit the allocation of one instance of the same micro-service per node. Thus, only one micro-service replica can be deployed on the same node. This constraint is shown in (10).

$$\forall a \,\varepsilon\, A, id \,\varepsilon\, ID, s \,\varepsilon\, S, \beta_i \,\varepsilon\, \beta : \sum_{n \,\varepsilon\, N} P^{a,id}_{s,\beta_i}(n) \le 1.0 \tag{10}$$

CPU and memory constraints have also been updated as shown in (11) and (12), respectively. Similarly, in (13), bandwidth limitations have been defined. Bandwidth requirements have been added to the model so that expected bandwidth demands can be met given the limited network link capacity.

$$\forall n \,\varepsilon\, N : \sum_{a \,\varepsilon\, A} \sum_{id \,\varepsilon\, ID} \sum_{s \,\varepsilon\, S} \sum_{\beta_i \,\varepsilon\, \beta} P^{a,id}_{s,\beta_i}(n) \times \omega_s \le \Omega_n \tag{11}$$

$$\forall n \,\varepsilon\, N : \sum_{a \,\varepsilon\, A} \sum_{id \,\varepsilon\, ID} \sum_{s \,\varepsilon\, S} \sum_{\beta_i \,\varepsilon\, \beta} P^{a,id}_{s,\beta_i}(n) \times \gamma_s \le \Gamma_n \tag{12}$$

$$\forall n \,\varepsilon\, N : \sum_{a \,\varepsilon\, A} \sum_{id \,\varepsilon\, ID} \sum_{s \,\varepsilon\, S} \sum_{\beta_i \,\varepsilon\, \beta} P^{a,id}_{s,\beta_i}(n) \times \delta_s \le \Delta_n \tag{13}$$

Secondly, several modifications have been made to the wireless formulations since one additional LPWAN technology has been included in the model. Sensors associate with gateways through an AID, a unique value assigned to a sensor by the gateway during association handshake. A constraint has been added to the model, ensuring that the AID limit on each gateway is respected. Therefore, by using the sensor association matrix $U_{sr,gw}$, the AID limitation can be expressed as shown in (14). The total number of AIDs attributed to a gateway must be lower than the total number of available AIDs.

$$\forall gw \,\varepsilon\, GW \;:\; \sum_{sr \,\varepsilon\, SR} \theta_{sr} \times U_{sr,gw} \leq \Theta_{gw} \tag{14}$$

A constraint is also added to ensure that sensors are connected with at least one gateway to be able to send the collected data. This constraint is represented by (15).

$$\forall sr \,\varepsilon\, SR \;:\; \sum_{gw \,\varepsilon\, GW} U_{sr,gw} \times A_{sr,gw} = 1 \tag{15}$$

Then, the association of sensors is based on the distance matrix $D_{gw,sr}$ and on the path loss matrix $PL_{gw,sr}$. On the one hand, for IEEE 802.11 ah, the path loss is calculated based on the path loss formula for urban macro deployments at a central frequency (f_c) of 900 MHz. This formulation can be expressed as in (16), with the distance d in meters. On the other hand, for LoRaWAN, the path loss is calculated based on the path loss formula presented in [37] for the Dortmund use case at a central frequency (f_c) of 868 MHz. This formulation can be expressed as in (17), with the distance d in kilometers.

$$PL(dB) = 8 + 37.6 \log_{10}(d) \tag{16}$$

$$PL(dB) = 132.25 + 10 \times 2.65 \log_{10}(d) \tag{17}$$

Thirdly, constraints have been added on network slicing. The main goal behind virtual slicing is to bring flexibility to the network by splitting the wireless traffic. Resources are reserved for each slice sl on the different gateways. Each slice sl is characterized by a bandwidth matrix $B_{sl,gw}$. The bandwidth matrix $B_{sl,gw}$ depends on the LPWAN technology and on the corresponding maximum bandwidth allowed for each sensor. For IEEE 802.11ah, the value of 256 Kbit/s has been considered, while for LoRaWAN, 50 Kbit/s has been chosen, since this is the theoretical maximum possible bandwidth for each sensor. The bandwidth matrix $B_{sl,gw}$ (in Mbit/s) can be expressed as in (18).

$$B_{sl,gw} = \begin{cases} \frac{\Theta_{gw} \times 0.256}{SL} & \text{if } I_{gw} = 1 \\ \frac{\Theta_{gw} \times 0.050}{SL} & \text{if } L_{gw} = 1 \end{cases} \tag{18}$$

The data rate matrix $\eta_{sr,sl,gw}$ (in Mbit/s) estimates the available bandwidth capacity for the sensor sr depending on the bandwidth assigned to the slice sl of gateway gw, which is given by (19).

$$\eta_{sr,sl,gw} = \begin{cases} 0.256 & \text{if } I_{gw} = 1 \\ SF \times \frac{B_{sl,gw}}{2^{SF}} & \text{if } L_{gw} = 1 \end{cases} \tag{19}$$

An additional constraint is added to ensure that sensors access the correspondent slice of their associated application. This constraint is represented by (20).

$$\forall sr \,\varepsilon\, SR, sl \,\varepsilon\, SL \;:\; U_{sr,sl} = A_{sr,sl} \tag{20}$$

To limit the association of sensors to only one slice from a given gateway, a constraint represented by (21) has been applied. Two constraints are also added to make sure that the formulation only selects one slice and one gateway for each sensor. These two constraints are shown in (22).

$$\forall sr \,\varepsilon\, SR, sl \,\varepsilon\, SL \;:\; \sum_{gw \,\varepsilon\, GW} U_{sr,sl,gw} = 1 \tag{21}$$

$$U_{sr,sl,gw} = \begin{cases} 1 & \text{if } U_{sr,sl} = 1 \wedge U_{sr,gw} = 1 \\ 0 & \text{if } U_{sr,sl} = 0 \vee U_{sr,gw} = 0 \end{cases} \tag{22}$$

To limit the traffic in each slice, a constraint given by (23) has been used to guarantee that the total wireless traffic on each slice sl does not exceed the maximum data rate capacity on each gateway gw.

$$\forall sl \,\varepsilon\, SL, gw \,\varepsilon\, GW \;:\; \sum_{sr \,\varepsilon\, SR} \eta_{sr,sl,gw} \times U_{sr,sl,gw} \leq B_{sl,gw} \tag{23}$$

Fourthly, constraints are added about the data transfer time T_{sr} and the gateway bandwidth factor ζ_{gw}. The data transfer time of a single message can be expressed by using the sensor's data rate $\eta_{sr,sl,gw}$ and the total number of bits to be transmitted π_{sr}. Thus, the data transfer time of each sensor sr is given by (24), in which the data transfer time is affected by ζ_{gw}. The gateway bandwidth factor ζ_{gw} is used to make sure that the formulation considers that a higher number of sensors associated to a single gateway will decrease the available bandwidth per sensor and, consequently, increase the data transfer time. The various constraints added to the model are shown in (25). These values are based on the work presented in [38]. Their experiments showed that increasing the number of sensors per gateway greatly decreases the maximum attainable data rate of each sensor.

$$\begin{gathered} \forall sr \,\varepsilon\, SR, sl \,\varepsilon\, SL, gw \,\varepsilon\, GW \;: \\ T_{sr} = \left(\frac{\pi_{sr} \times 1000}{\eta_{sr,sl,gw}} \right) \times \zeta_{gw} \quad \text{(in ms)} \end{gathered} \tag{24}$$

$$\zeta_{gw} = \begin{cases} 1.0 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : U_{sr,gw} \leq 3 \\ 1.11 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 3 < U_{sr,gw} \leq 5 \\ 1.25 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 5 < U_{sr,gw} \leq 8 \\ 1.43 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 8 < U_{sr,gw} \leq 12 \\ 1.67 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 12 < U_{sr,gw} \leq 15 \\ 2.0 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 15 < U_{sr,gw} \leq 18 \\ 2.5 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 18 < U_{sr,gw} \leq 26 \\ 3.33 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 26 < U_{sr,gw} \leq 33 \\ 5.0 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : 33 < U_{sr,gw} \leq 40 \\ 10.0 & \text{if } \sum_{sr,gw\,\varepsilon\, SR,GW} : U_{sr,gw} \geq 40 \end{cases} \tag{25}$$

Finally, constraints are added about the user latency ψ_u. The user latency corresponds to the propagation time (in ms) of a request from the user u to reach an instance of the last micro-service in the application's service chain with which user u is connected to, as expressed by (26). As shown, the user latency depends on which node the last micro-service replica is allocated. If the micro-service instance is deployed on a node far from the user, the user latency will thus be higher. Consequently, the E2E latency is two times the value of ψ_u since it represents the time it takes for a user request to reach the last service in the chain and coming back to the user.

$$\begin{aligned} &\forall u\, \varepsilon\, U, a\, \varepsilon\, A, id\, \varepsilon\, ID, s\, \varepsilon\, S, \beta_i\, \varepsilon\, \beta, n\, \varepsilon\, N \; : \\ &\textbf{if } G_{u,a,id} = 1 \quad (u \text{ associated with app. } a \text{ with SFC id. } id) \\ &\textbf{if } \iota_s \times U^{u,a,id}_{s,\beta_i}(n) = 1 \quad (u \text{ connected with last instance}) \\ &\textbf{Then } \psi_u = \tau_{u,n} \quad (\text{in ms}) \end{aligned} \tag{26}$$

3.3.2. *Minimizing Service Migrations - (MIN M)*

Although this objective has already been considered in the previous version of the model, the service migration estimation in the model has been reformulated. In a dynamic use case, micro-service replicas may need to be reallocated from one node to another to provide the optimal provisioning solution. However, it may be desirable to find a sub-optimal solution, in which service migrations are kept to a minimum to reduce the delay caused by service reallocations. Since the model is executed iteratively, the placement matrix from the previous iteration P^{i-1} is added to the model, which is compared with the current placement matrix P to reduce the service migrations needed to achieve the next objective. The decision variable $M^{a,id}_{s,\beta_i}(n)$ is used to determine the correspondent service migrations as shown by (27).

$$\forall a\, \varepsilon\, A, id\, \varepsilon\, ID, s\, \varepsilon\, S, \beta_i\, \varepsilon\, \beta, n\, \varepsilon\, N \; :$$
$$M^{a,id}_{s,\beta_i}(n) = \begin{cases} 0 & \begin{array}{l} \text{if} \quad P^{a,id}_{s,\beta_i}(n) = 1 \wedge P^{a,id}_{s,\beta_i}(n)^{i-1} = 1 \\ \text{if} \quad P^{a,id}_{s,\beta_i}(n) = 0 \wedge P^{a,id}_{s,\beta_i}(n)^{i-1} = 0 \end{array} \\ 1 & \text{Otherwise} \end{cases} \tag{27}$$

Then, the service migrations matrix $M^{a,id}_{s,\beta_i}$ is calculated as given by (28).

$$\forall a\, \varepsilon\, A, id\, \varepsilon\, ID, s\, \varepsilon\, S, \beta_i\, \varepsilon\, \beta \; :$$
$$M^{a,id}_{s,\beta_i} = \begin{cases} 1 & \text{if} \quad \sum_{n\,\varepsilon\, N} M^{a,id}_{s,\beta_i}(n) \geq 1 \\ 0 & \text{if} \quad \sum_{n\,\varepsilon\, N} M^{a,id}_{s,\beta_i}(n) = 0 \end{cases} \tag{28}$$

Thus, the minimization of service migrations compared to the previous iteration can be expressed as shown in (29).

$$min \sum_{a\,\varepsilon\, A} \sum_{id\,\varepsilon\, ID} \sum_{s\,\varepsilon\, S} \sum_{\beta_i\,\varepsilon\, \beta} M^{a,id}_{s,\beta_i} \tag{29}$$

In most cases, this objective will obtain solutions where service migrations are not admitted at all (i.e. 0% service migrations). To allow the model to achieve intermediate solutions based on a predefined limit on the maximum number of allowed service migrations, a constraint has been added to the model based on the service migration factor μ. This constraint is given by (30).

$$\forall a\, \varepsilon\, A, id\, \varepsilon\, ID \; :$$
$$\sum_{s\,\varepsilon\, S} \sum_{\beta_i\,\varepsilon\, \beta} M^{a,id}_{s,\beta_i} \times I_{a,s} \leq \mu \times \sum_{s\,\varepsilon\, S} I_{a,s} \times \beta_{a,id,s} \tag{30}$$

3.3.3. *Minimizing Active Nodes - (MIN N)*

This objective concerns the minimization of the number of nodes used in the service allocation, which results in cost and energy savings. This optimization can be expressed as shown in (31) by using the node utilization matrix U_n.

$$min \sum_{n\,\varepsilon\, N} U_n \tag{31}$$

3.3.4. *Minimizing Active Gateways - (MIN GW)*

This optimization aims to minimize the number of active gateways in the network. This objective ensures that a minimum number of gateways is used to provide connectivity to all the sensors. This results in energy and cost savings, however, it increases the data transfer time for each sensor. A clear trade-off exists between this objective and the minimization of the sensor's data transfer time. Service providers may opt for one of the two strategies, depending on their service characteristics and the network behavior at a given moment. By using the gateway utilization matrix U_{gw}, the minimization can be expressed as shown in (32).

$$min \sum_{gw\,\varepsilon\, GW} U_{gw} \tag{32}$$

3.3.5. *Minimizing Network Latency - (MIN NL)*

This objective is related to latency reduction in the communication between micro-services from the same application corresponding to the proper service chain path. This is expressed by the Flow Factor $\Upsilon s_i, s_j$ shown in (33). Thus, by allocating each micro-service as close to the next micro-service in the service chain as possible, the latency is reduced. The SFC latency matrix $\Lambda_{a,id}$ is determined by using the Flow matrix F as stated in (34).

$$\Upsilon s_i, s_j = I_{a,s_i} \times I_{a,s_j} \times \alpha_{s_i,s_j} \tag{33}$$

$$\forall a \; \varepsilon \; A, id \; \varepsilon \; ID : \Lambda_{a,id} = \sum_{s_i \, \varepsilon \, S} \sum_{\beta_i \, \varepsilon \, \beta} \sum_{s_j \, \varepsilon \, S} \sum_{\beta_j \, \varepsilon \, \beta} \sum_{n_1 \, \varepsilon \, N} \sum_{n_2 \, \varepsilon \, N} \Upsilon s_i, s_j \times \tau_{n_1,n_2} \times F^{a,id,s_i,\beta_i}_{s_j,\beta_j}(n_1, n_2) \tag{34}$$

The Flow matrix F is also subjected to various constraints to accurately represent network flows, specifically in terms of flow conservation, which ensures no flow is lost within the network, as shown in (35).

$$\forall a \; \varepsilon \; A, id \; \varepsilon \; ID, s_i \; \varepsilon \; S, \beta_i \; \varepsilon \; \beta, s_j \; \varepsilon \; S, \beta_j \; \varepsilon \; \beta : \sum_{n_1 \, \varepsilon \, N} \sum_{n_2 \, \varepsilon \, N} F^{a,id,s_i,\beta_i}_{s_j,\beta_j}(n_1, n_2) = \begin{cases} 1 & \text{if } P^{a,id}_{s_i,\beta_i}(n_1) = 1 \wedge P^{a,id}_{s_j,\beta_j}(n_2) = 1 \\ 0 & \text{Otherwise} \end{cases} \tag{35}$$

Thus, this objective can be expressed as shown in (36).

$$min \sum_{a \, \varepsilon \, A} \sum_{id \, \varepsilon \, ID} \Lambda_{a,id} \tag{36}$$

3.3.6. Minimizing Data Transfer Time - (MIN T)

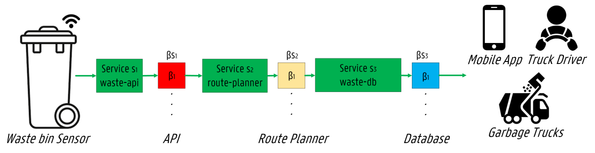
In an IoT scenario, sensors need to communicate with gateways to send their collected data to the Fog-cloud infrastructure. This objective concerns the reduction of the data transfer time of a single message by optimizing the sensor's association with the several heterogeneous gateways available in the network. If a given gateway is supporting a high number of sensors, the maximum bandwidth for each sensor drops and thus the data transfer time increases. In the model, the data transfer time T_{sr} is affected by the gateway bandwidth factor ζ_{gw}, which depends on the number of associated sensors with a given gateway. This minimization can be expressed as shown in (37).

$$min \sum_{sr \, \varepsilon \, SR} T_{sr} \tag{37}$$

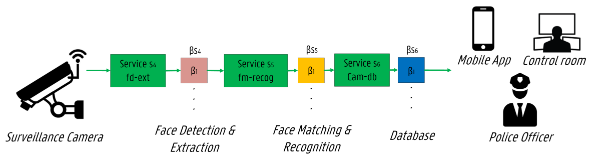
3.3.7. Minimizing User latency - (MIN UL)

Previously, two optimization objectives regarding latency have been presented. However, neither objective addresses the expected latency for each user based on their assigned application. To fully address user latency, the model needs to make decisions not only in terms of micro-service allocation but also on the replication factor for each micro-service. If users are spread across the network, it may be beneficial to allocate more micro-service instances to reduce the latency expected by each user. All these factors should be taken into account to fully achieve the minimization of the user latency. In the model, the user latency is between the user and the last service in the chain, since it is expected that users access the last service to obtain the required information. This objective can be expressed as shown in (38) based on the previously presented constraints regarding user latency.

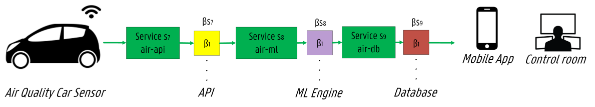
$$min \sum_{u \, \varepsilon \, U} \psi_u \tag{38}$$



(a) Waste Management Use Case.



(b) Surveillance Camera Use Case.



(c) Air Quality Monitoring Use Case.

Figure 1: The container-based Service Function Chains envisioned for the three evaluated use cases.

4. Use Cases

In this section, three Smart City use cases within the scope of Antwerp's City of Things testbed [39] are introduced. First, a Waste Management use case is presented where sensors are installed in waste bins to collect bins' fill levels to optimize garbage collection through route optimization services. Then, a surveillance camera scenario is detailed where cameras are placed in crowded streets to send continuous video streams to Fog locations where face detection and recognition services are applied in a distributed manner. Finally, an air quality monitoring use case is described where sensors are installed on vehicles to collect air quality data and then alert citizens in case air pollution levels are detected through machine learning (ML) services. In Fig. 1, the container-based service chains for the three use cases are illustrated and the correspondent deployment requirements are shown in Table 4.

Table 4
Deployment properties of the three evaluated use cases.

Application	Service	Chain Position	CPU (cpu)	RAM (Mi)	Min. Band. (Mbit/s)	Max. Users	User Cost	Device Upload Message Size (in B)
Waste (a_1)	waste-api (s_1)	1	0.25 (ω_{s_1})	0.25 (γ_{s_1})	5.0 (δ_{s_1})	5 (ρ_{s_1})	0.25 (λ_u)	37 (π_a = 296 bits)
	route-planner (s_2)	3	0.5 (ω_{s_2})	1.0 (γ_{s_2})	8.0 (δ_{s_2})	8 (ρ_{s_2})		
	waste-db (s_3)	2	0.5 (ω_{s_3})	1.0 (γ_{s_3})	5.0 (δ_{s_3})	5 (ρ_{s_3})		
Camera (a_2)	fd-ext (s_4)	1	0.5 (ω_{s_4})	0.5 (γ_{s_4})	4.0 (δ_{s_4})	4 (ρ_{s_4})	1.0 (λ_u)	1500 (π_a = 12000 bits)
	fm-recog (s_5)	2	1.0 (ω_{s_5})	2.0 (γ_{s_5})	8.0 (δ_{s_5})	8 (ρ_{s_5})		
	cam-db (s_6)	3	0.5 (ω_{s_6})	0.5 (γ_{s_6})	5.0 (δ_{s_6})	5 (ρ_{s_6})		
Air (a_3)	air-api (s_7)	1	0.25 (ω_{s_7})	0.25 (γ_{s_7})	4.0 (δ_{s_7})	4 (ρ_{s_7})	0.5 (λ_u)	93 (π_a = 744 bits)
	ml-engine (s_8)	2	0.5 (ω_{s_8})	1.0 (γ_{s_8})	8.0 (δ_{s_8})	8 (ρ_{s_8})		
	air-db (s_9)	3	0.5 (ω_{s_9})	0.5 (γ_{s_9})	4.0 (δ_{s_9})	4 (ρ_{s_9})		

4.1. Waste Management Scenario

Although waste bins are located everywhere (e.g. restaurants, office buildings, retail stores), garbage collection has been an inefficient service for years. Garbage trucks follow a predefined route without knowing if waste bins are currently empty or full. Furthermore, waste bins may get overloaded before the planned collection. This traditionally results in high maintenance and fuel costs. However, IoT can improve the performance of waste collection by gathering bin data [40]. For instance, sensors can be installed into waste bins to monitor which bins are full. By sending the collected data to a Fog-cloud infrastructure, route planning services can be executed to find the optimal route for each truck based on the bins' fill levels. Thus, drivers do not waste time driving to empty bins and broken bins may be repaired quickly. Drivers can access their optimized route through a mobile application that connects to a database service, enabling them to improve their customer service. Therefore, an IoT-based waste management service provides a more efficient waste collection through route optimization and higher driver productivity. The objective of this use case is to enable access to waste bin information.

Considering that for the waste management use case, each upload message is composed of a string of 12 chars (GPS Location - geohash) equal to 12 bytes, a 32-bit integer (timestamp) equal to 4 bytes and 1 floating point 64-bit number (fill level measure) equal to 8 bytes, the total number of payload bytes to be transmitted from the sensor to the Fog-cloud infrastructure is 24 bytes. In our evaluation, a general 13-byte header has been considered in each message as in LoRaWAN technologies. Therefore, each upload message is transmitted with at least 37 bytes, which is equal to 296 bits.

4.2. Surveillance Camera Scenario

Surveillance services have become highly relevant due to the possibility of identifying individuals or even objects in crowded areas [41]. However, connectivity issues still need to be addressed, including data transfer over limited bandwidth and high latency in sensor-cloud communications. For instance, imagine a low-quality surveillance camera requiring a continuous streaming bandwidth of 256 Kbit/s. Sending the entire data from a single video camera to the cloud translates into approximately 0.08 TB/monthly. To reduce the amount of data transmitted to the cloud, Fog-cloud infrastructures may be adopted by performing data operations locally. Surveillance cameras placed on particular streets or crowded areas send continuous video streams to Fog locations, where face recognition algorithms are performed in a distributed manner. Thus, suspicious individuals can be detected in near real-time. Fog nodes located close to the surveillance cameras receive their video streams and perform a first-level analysis, such as face detection and feature extraction tasks. Then, Fog nodes send the results to the cloud for global analysis operations, such as face matching and recognition operations, due to its high computational requirements. Afterwards, global outcomes can be presented in a central dashboard in a control room. Also, police officers may access the detection results through a mobile application. This approach has been previously presented in [42], to enable anomaly detection services in Fog Computing. An IoT-based surveillance camera service enables an efficient way of recognizing individuals in crowded areas by distributing tasks between Fog and cloud. The objective of this use case is to provide a near real-time face detection system.

Considering that the surveillance camera use case is a video streaming scenario, each upload message is composed of 1500 bytes, equal to 12000 bits. The value of 1500 bytes has been selected due to the Maximum Transmission Unit (MTU) limitation, which defines the largest packet size that can be sent over a network connection.

4.3. Air Quality monitoring scenario

Air pollution has become the largest environmental and public health challenge in the world. Air pollution leads to adverse effects on human health and climate change. As an initial proof of concept within Antwerp's City of Things project, air quality sensors have been installed on vehicle rooftops in collaboration with the Belgian postal services, Bpost. For daily mail delivery, Bpost keeps cars continuously driving around the city of Antwerp, thus enabling air quality sensors gathering data with broad city coverage. These sensors collect measures of typical gases (e.g. carbon dioxide (CO_2), Nitrogen Dioxide (NO_2), Particulate Matter indicators (PMIs)), and also climate data, such as temperature and humidity, which are then annotated with GPS locations. By sending the collected data to a Fog-cloud infrastructure [43], ML services can be executed to find patterns and anomalies in the data. Then, citizens can be alerted in case unhealthy levels of air pollution are detected through a mobile application in near real-time. Thus, an IoT-based air quality monitoring system enables the detection of high concentrations of organic compounds and contributes to improved public health [44].

Considering that for the air quality monitoring use case, each upload message is composed of a string of 12 chars (GPS Location - geohash) equal to 12 bytes, a 32-bit integer (timestamp) equal to 4 bytes, 3 floating-point 64-bit numbers (PMIs) equal to 24 bytes and 5 floating-point 64-bit numbers (Nitrogen, Ozone, Carbon Monoxide, Temperature, Humidity) equal to 40 bytes, the total number of payload bytes to be transmitted from the sensor to the Fog-cloud infrastructure is 80 bytes. Therefore, each upload message is transmitted with at least 93 bytes, which is equal to 744 bits.

5. Evaluation Setup

In this section, the network infrastructure used for the MILP model is described. Then, input variables and optimization policies used in the evaluation are shown.

5.1. Network Infrastructure

Table 5
Input variables for the MILP model.

Variables	Value
A	3
S	9
L	5
SL	3
GW	35
$I_{a,s}$	$a_1 : s_1 \rightarrow s_2 \rightarrow s_3$ $a_2 : s_4 \rightarrow s_5 \rightarrow s_6$ $a_3 : s_7 \rightarrow s_8 \rightarrow s_9$
β	10 (Maximum Replication factor)
L_{gw}	5 (LoRaWAN gateways)
I_{gw}	30 (IEEE 802.11ah gateways)

The Fog-cloud infrastructure illustrated in Fig. 2 has been represented in the model. Input variables are presented in Table 5 based on the described network infrastructure. A total area of 324 km^2 has been considered. The Fog-cloud infrastructure is deployed on five locations L, where the micro-service provisioning is possible. Each location manages a set of three nodes. The hardware configurations of each node are shown in Table 6. Each node, gateway, and sensor have a given location associated. Coordinate positions (x, y) are randomly attributed to each sensor, while for each gateway, coordinate positions are strategically attributed to cover the entire five locations. Based on these coordinates, the distance matrix D is calculated by the euclidean distance formula as shown in (39). Then, the path loss matrix PL is calculated based on the path loss formulas previously described by using the calculated distance matrix values. The bandwidth matrix B_{n_1,n_2} is based on the available bandwidth capacity. Also, all latency matrices τ are calculated based on the shown latency values.

Table 6
The hardware configurations of each node.

Node	CPU (cpu)	RAM (Mi)	Band. (Mbit/s)
Worker 1 (n_1)	2.0 (Ω_{n_1})	4.0 (Γ_{n_1})	10.0 (Δ_{n_1})
Worker 2 (n_2)	2.0 (Ω_{n_2})	4.0 (Γ_{n_2})	10.0 (Δ_{n_2})
Worker 3 (n_3)	1.0 (Ω_{n_3})	2.0 (Γ_{n_3})	5.0 (Δ_{n_3})
Worker 4 (n_4)	2.0 (Ω_{n_4})	4.0 (Γ_{n_4})	10.0 (Δ_{n_4})
Worker 5 (n_5)	1.0 (Ω_{n_5})	2.0 (Γ_{n_5})	5.0 (Δ_{n_5})
Worker 6 (n_6)	2.0 (Ω_{n_6})	4.0 (Γ_{n_6})	10.0 (Δ_{n_6})
Worker 7 (n_7)	2.0 (Ω_{n_7})	4.0 (Γ_{n_7})	10.0 (Δ_{n_7})
Worker 8 (n_8)	2.0 (Ω_{n_8})	4.0 (Γ_{n_8})	10.0 (Δ_{n_8})
Worker 9 (n_9)	1.0 (Ω_{n_9})	2.0 (Γ_{n_9})	5.0 (Δ_{n_9})
Worker 10 (n_{10})	2.0 ($\Omega_{n_{10}}$)	4.0 ($\Gamma_{n_{10}}$)	10.0 ($\Delta_{n_{10}}$)
Worker 11 (n_{11})	2.0 ($\Omega_{n_{11}}$)	4.0 ($\Gamma_{n_{11}}$)	5.0 ($\Delta_{n_{11}}$)
Worker 12 (n_{12})	2.0 ($\Omega_{n_{12}}$)	4.0 ($\Gamma_{n_{12}}$)	10.0 ($\Delta_{n_{12}}$)
Worker 13 (n_{13})	6.0 ($\Omega_{n_{13}}$)	16.0 ($\Gamma_{n_{13}}$)	30.0 ($\Delta_{n_{13}}$)
Worker 14 (n_{14})	6.0 ($\Omega_{n_{14}}$)	16.0 ($\Gamma_{n_{14}}$)	30.0 ($\Delta_{n_{14}}$)
Master (n_{15})	8.0 ($\Omega_{n_{15}}$)	24.0 ($\Gamma_{n_{15}}$)	30.0 ($\Delta_{n_{15}}$)

$$D(gw, sr) = \sqrt{(x_{gw} - x_{sr})^2 + (y_{gw} - y_{sr})^2} \tag{39}$$

5.2. Input variables & Optimization Policies

Table 7
The evaluated optimization policies.

A - Minimum Latency	*B* - Energy Efficiency
1 - MAX R	1 - MAX R
2 - MIN UL	2 - MIN N
3 - MIN NL	3 - MIN GW
4 - MIN T	

The described MILP formulation has been implemented in Java using the IBM ILOG CPLEX ILP solver [45]. As previously mentioned, several optimization objectives are executed iteratively, where the optimal solutions of previous objectives are added as additional constraints to the model for the subsequent iterations. Two policies have been evaluated. In each iteration of the model, a different optimization

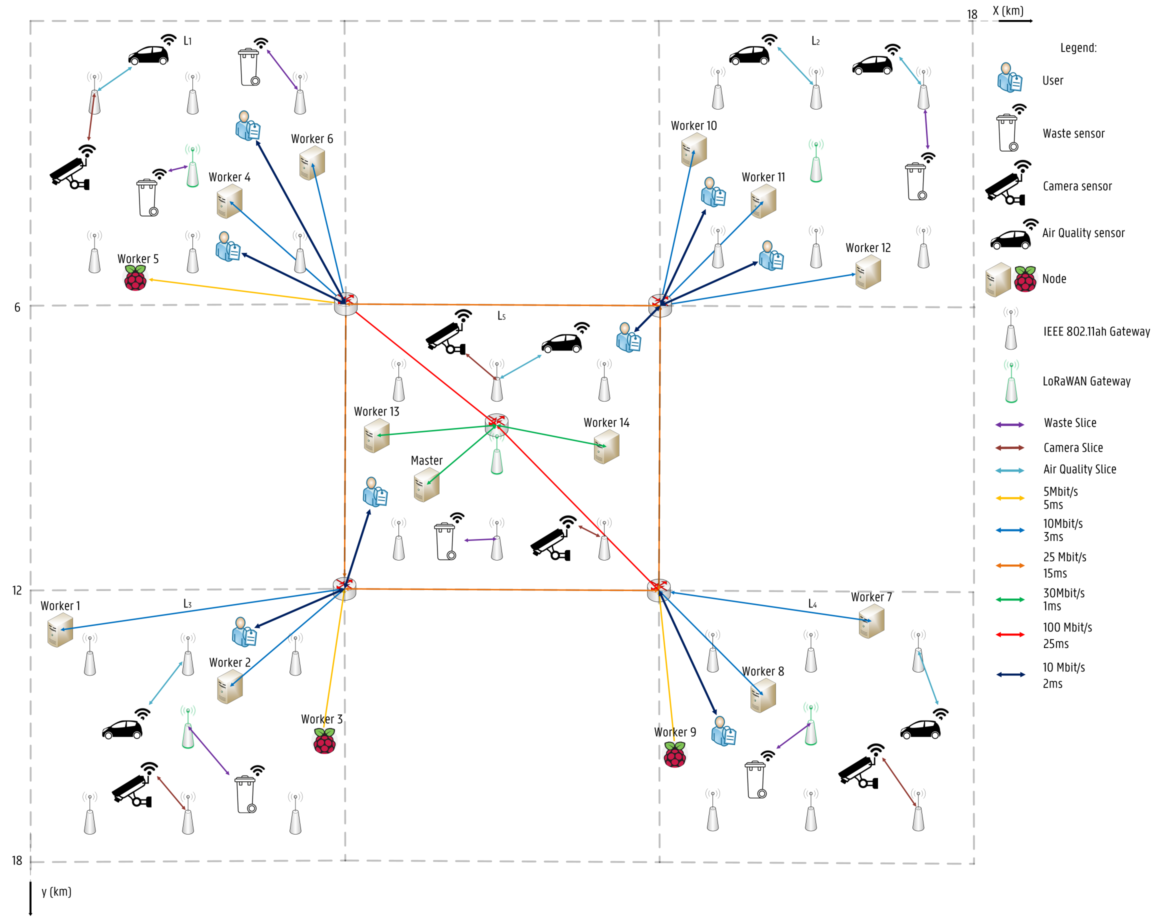


Figure 2: A Fog-cloud infrastructure for the IoT service allocation problem.

objective has been considered. In Table 7, the evaluated optimization policies are shown. On the one hand, policy *A* is responsible for finding micro-service allocations focused on reducing latency. Also, it tries to minimize the sensor's data transfer time by spreading wireless traffic across multiple gateways. On the other hand, policy *B* is related to energy efficiency, since it minimizes the number of active nodes needed to allocate all the required micro-service instances. Furthermore, policy *B* maximizes gateway usage by connecting all sensors to the minimum number of gateways possible. It should be highlighted that for the two policies, the number of accepted user requests is maximized first. The model has been executed on the HPC-UGent supercomputing infrastructure [46]. Cluster nodes have been requested to run the experiments (2 x 18-core Intel Xeon Gold 6140 @ 2.3 GHz processor with 32 GB of memory). All policies have been evaluated 30 times and confidence intervals of 95% have been considered in the evaluation.

6. Results

In this section, the results are detailed. First, the execution time of the different optimization objectives is shown, followed by performance outcomes on the three presented use cases. Finally, a joint use case is shown followed by an

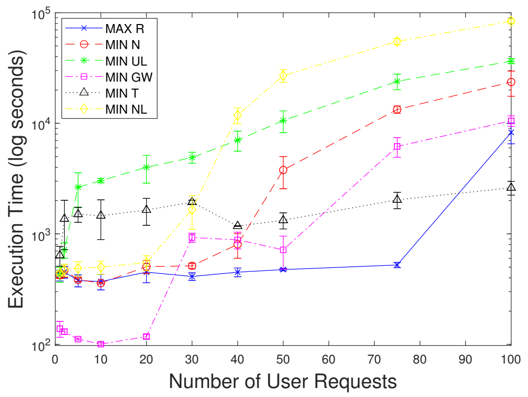


Figure 3: The execution time of the different optimization objectives for the waste management use case.

analysis concerning the impact of service migrations when provisioning strategies are altered.

6.1. Execution Time per Objective

In Fig. 3, the execution time of the different optimization objectives is shown for the waste management use case. The MILP model has been executed until the optimal solution for each objective has been found. The number of sensors has been set to 100 throughout the experiment. By increasing the number of user requests, the execution time of most objectives increases due to the increased allocation complexity. More micro-service instances need to be allocated since more users are sending requests. The most affected objectives by the increase of user requests are the *MIN UL* and the *MIN NL*. The complexity increases significantly for these two objectives since several users are spread across the network area and the latency needs to be reduced. In fact, the *MIN UL* requires on average 43 minutes to find the optimal solution already for 5 user requests while the *MIN NL* requires on average 8 minutes for the same number of user requests. Additionally, the execution time of the *MIN N* objective significantly increases for user requests higher than 40. Over 40 requests, still minimizing the number of nodes used in the micro-service allocation becomes a difficult task. For example, the *MIN N* requires on average 13 minutes and 63 minutes to find the optimal solution for 40 and 50 user requests, respectively. Also, the execution time of both *MIN T* and *MIN G* objectives slightly increase during the experiment since the number of sensors is kept constant and, thus, the complexity of both objectives does not increase significantly. The *MAX R* objective slightly increases throughout of the experiment, however, it greatly increases for 100 user requests requiring more than an hour and a half to find the optimal solution, meaning that finding any solution that meets all constraints and maximizes the acceptance of 100 requests is quite time-consuming. Additionally, it should be highlighted that regarding sequential convergence speed, the optimization order that is applied has a strong impact on execution time and on the solution space. For instance, the two strategies evaluated can be combined into a single optimization strategy. As a first objective, *MIN N* is applied followed by the *MIN UL* optimization. As expected, the second optimization is quite limited since the overall value found in the first iteration is added as a constraint. So, the optimization in terms of user latency only occurs if the current allocation scheme can be further optimized without changing the overall value found in the first iteration. Thus, the execution time of the second objective is significantly faster when the first objective is considered since user latency cannot be fully optimized.

In summary, the MILP model can find a solution for most objectives in rather small execution time for user requests lower than 20 (i.e most cases 4 - 10 minutes). However, for the MIN UL, the model takes on average 1 hour and 6 minutes to find a solution for 20 user requests. Both *MIN UL* and *MIN NL* objectives related to latency reduction do not scale well. For example, the *MIN UL* requires on average 4 hours and 18 minutes to find the optimal solution for 50 user requests while the *MIN NL* requires on average 14 hours and 15 minutes for the same number of user requests. The execution time of the other objectives is still acceptable for values lower than 75 user requests.

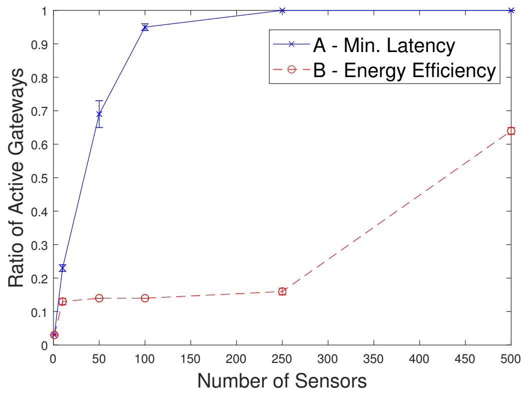


Figure 4: The gateway usage rate for the waste management use case.

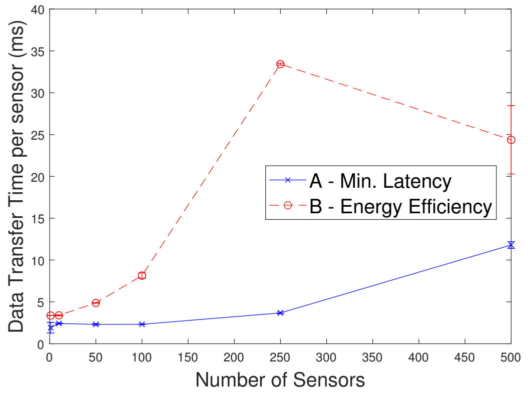


Figure 5: The data transfer time per sensor for the waste management use case.

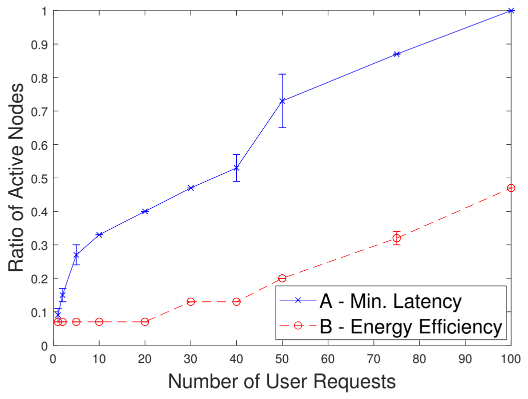


Figure 6: The node utilization rate for the waste management use case.

6.2. Waste Management Use Case

In Fig. 4, the ratio of active gateways for each policy is shown. On the one hand, high usage rates are shown for policy ***A*** since no optimization objective is included regarding gateway efficiency. On the other hand, for policy ***B***, the ratio of active gateways slightly increases with the increase of connected sensors. For example, for policy ***B***, results show that only 16% of the gateways are needed for 250 sensors and that only for 500 sensors, the ratio is higher than 60%. In

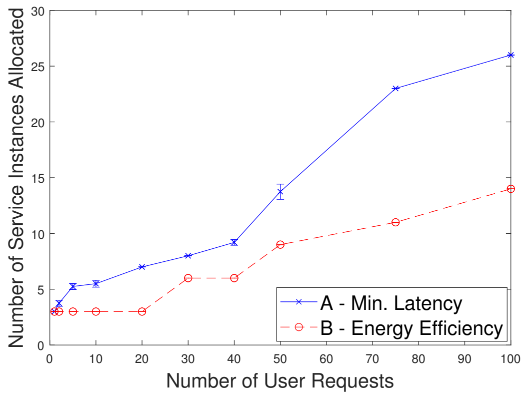


Figure 7: The number of micro-service instances allocated for the waste management use case.

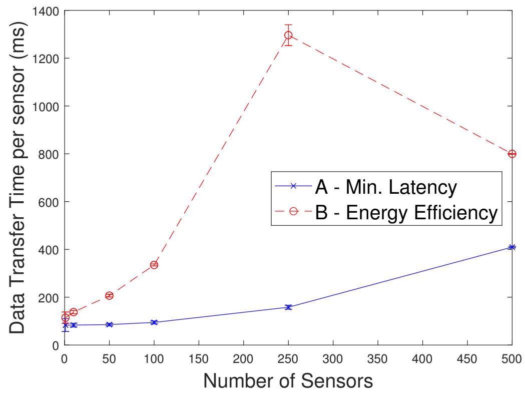


Figure 9: The sensor's data transfer time for the surveillance camera use case.

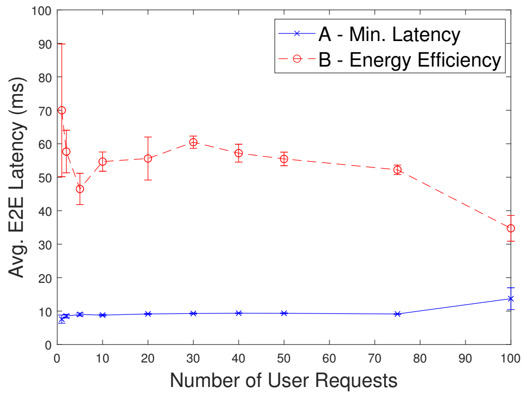


Figure 8: The expected E2E latency per user for the waste management use case.

contrast, policy A requires on average more than 90% of active gateways already for 100 sensors. Thus, policy B shows a higher gateway efficiency. In Fig. 5, the sensor's data transfer time for each policy is illustrated. As expected, policy A achieved the lowest values of data transfer time per sensor while policy B achieved the highest values since both policies represent opposing strategies. For example, for 250 sensors, each sensor requires on average 33 ms to send a message for policy B while for policy A, each sensor requires only 3.5 ms. This difference occurs because of the gateway usage rate. As previously shown, policy B needs on average only 16% active gateways while policy A requires 90% to minimize the sensor's data transfer time. In Fig. 6, the ratio of active nodes for each policy is shown. On the one hand, for policy B, the active number of nodes slightly increases with the increase of user requests, since this policy is focused on maximizing energy efficiency in the infrastructure. On the other hand, for policy A, the active nodes greatly increase since this policy is focused on reducing latency. It should be noted that for 40 and 50 user requests, the confidence interval for policy A is higher because user latency can be reduced by activating a different number of nodes, which depends on the user's location. This can be considered a borderline case, where a single user on a particular location could mean that another node needs to be active and thus the higher confidence interval. In Fig. 7, the exact number of micro-service instances allocated on the network for both policies is depicted. As expected, policy A allocates more micro-service instances on the network than policy B. This results in reduced E2E latency as shown in Fig. 8. Latency values between 8 and 10 ms are obtained for policy A throughout the experiment even for a high number of user requests while values between 30 and 70 ms are obtained for policy B since the micro-service allocation is not focused on E2E latency but energy efficiency. However, for 100 user requests, policy A is not able to further minimize latency since the infrastructure is already exhausted. Thus, the average E2E latency increases to 13 ms, which is still considerably low when compared to policy B, which obtains latency values of 34.73 ms.

In summary, the Waste Management scenario represents a low bandwidth use case. As shown, differences can be achieved in terms of micro-service allocation depending on the provisioning strategy. Node usage rates are distinct among the two evaluated strategies, which resulted in different results in terms of E2E latency. The gateway usage rate directly affects the data transfer time of each sensor, which could be an important factor. Nevertheless, the sensor's data transfer time variations for the two evaluated strategies (between 33 ms - 3.5 ms in the worst case) can be classified as not particularly relevant for this use case.

6.3. Surveillance Camera Use Case

The variations of the ratio of active gateways for both policies are similar to the waste management use case, thus the graph is not included to avoid repetition. Nevertheless, the sensor's data transfer time variations are higher as illustrated in Fig. 9 since this use case is related to a video streaming scenario (i.e. high bandwidth requirements). As shown, policy A achieved the lowest values of data transfer time per sensor. For instance, for 250 sensors, each sensor requires on average 158 ms to send a message for policy A while for policy B, each sensor requires at least 1.3 seconds. This significant difference occurs due to the high bandwidth requirements for this use case. In Fig. 10, the ratio of active nodes for each policy is shown. The ratio of active nodes for both

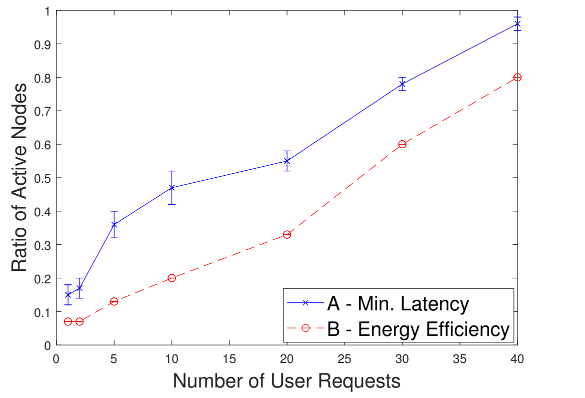


Figure 10: The node utilization rate for the surveillance camera use case.

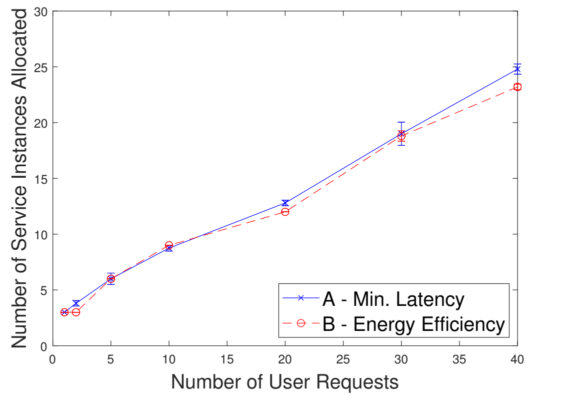


Figure 11: The number of micro-service instances allocated for the surveillance camera use case.

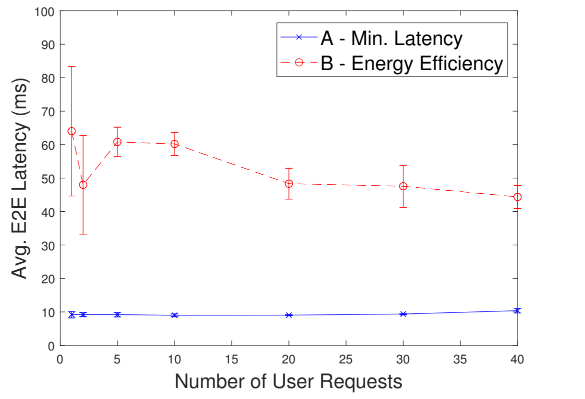


Figure 12: The expected E2E latency per user for the surveillance camera use case.

policies slightly increases with the increase of user requests. The difference between policy A and B is lower when compared to the waste management use case, due to the difference in user cost (λ_u), 1.0 and 0.25, respectively for both use cases. Also, the number of micro-service instances allocated on the network demonstrated in Fig. 11 clearly shows that the fluctuations are attenuated due to the user cost. Nevertheless, the pattern in terms of E2E latency remains similar as shown in Fig. 12. Latency values between 8 and 10 ms

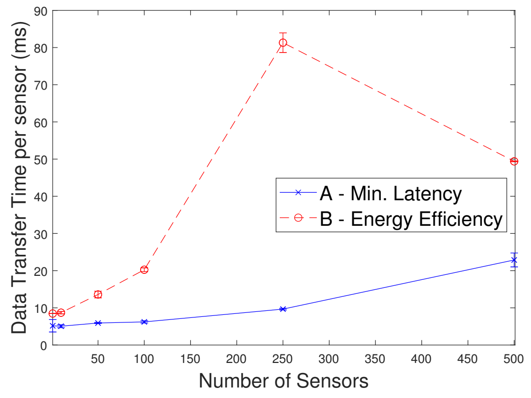


Figure 13: The sensor's data transfer time for the air quality monitoring use case.

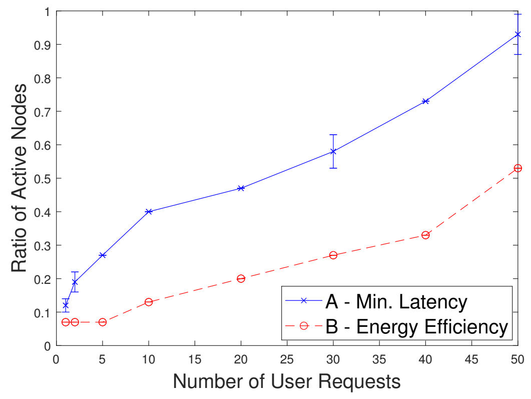


Figure 14: The node utilization rate for the air quality monitoring use case.

are obtained for policy A while values between 30 and 70 ms are obtained for policy B.

In summary, the Surveillance Camera scenario represents a high bandwidth use case regarding video streaming. As shown, significant differences are achieved in terms of micro-service allocation and the sensor's data transfer time. For instance, notable data transfer time variations (between 158 ms - 1.3 seconds in the worst case) have been shown, which make this use case extremely challenging in the IoT ecosystem.

6.4. Air Quality Monitoring Use Case

As expected, the variations of the ratio of active gateways for both policies are similar to the waste management and the surveillance use case, thus the graph is not included. In Fig. 13, the sensor's data transfer time for each policy is illustrated. For instance, for 250 sensors, each sensor requires on average 9.6 ms to send a message for policy A while for policy B, each sensor requires 81 ms. In Fig. 14, the ratio of active nodes for each policy is shown. The ratio of active nodes slightly increases for policy B with the increase of user requests while for policy A, the ratio significantly increases to minimize the user latency. For example, for 10 user requests, 40% of nodes are active for policy A while for policy B only 13.3% of nodes are needed. In Fig. 15,

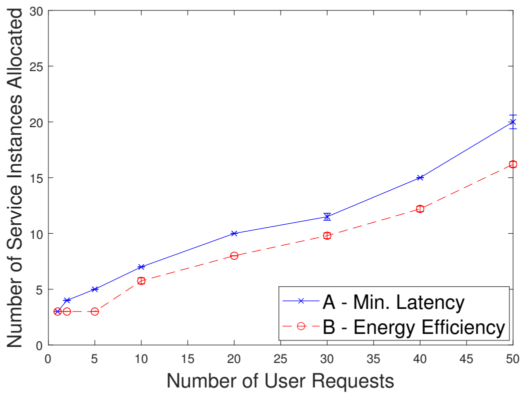


Figure 15: The number of micro-service instances allocated for the air quality monitoring use case.

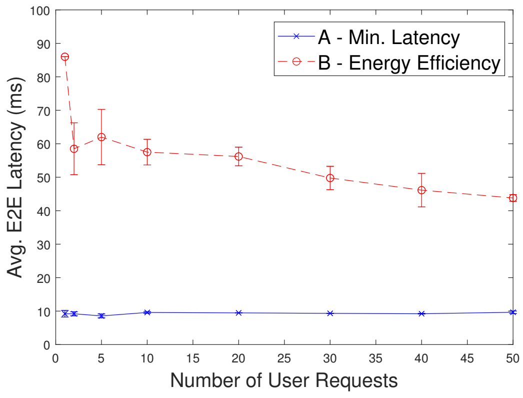


Figure 16: The expected E2E latency per user for the air quality monitoring use case.

the number of micro-service instances allocated on the network is shown. Similar to the surveillance scenario, the differences in terms of micro-service allocation are attenuated due to the user cost. Finally, in Fig. 16, the expected E2E latency is shown. As previously demonstrated for the other two scenarios, the E2E pattern remains the same. Latency values between 8 and 10 ms are obtained for policy *A* while values between 40 and 80 ms are obtained for policy *B*.

In summary, the Air Quality Monitoring scenario represents a medium bandwidth use case regarding a near real-time air pollution detection service. As shown, differences are achieved for the two evaluated strategies, specifically in terms of node usage rates and E2E latency. Also, variations in the sensor's data transfer time are obtained depending on the applied strategy. Results show differences between 9.6 ms - 81 ms, which make this use case more challenging than the waste management use case.

6.5. Joint Use Case: Optimizing all Use Cases together

The Joint use case corresponds to a real-world scenario where multiple applications need to be deployed in the network. The three previously evaluated use cases need to be allocated in the network at the same time since users need to access all their respective applications. The user's assigned

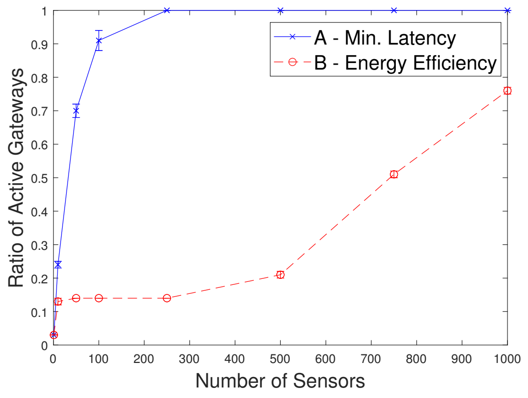


Figure 17: The gateway usage rate for the joint use case.

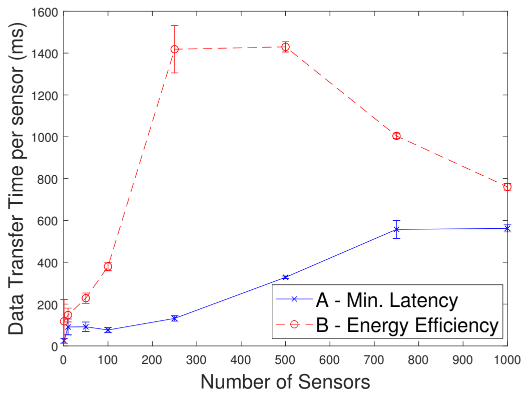


Figure 18: The sensor's data transfer time for the joint use case.

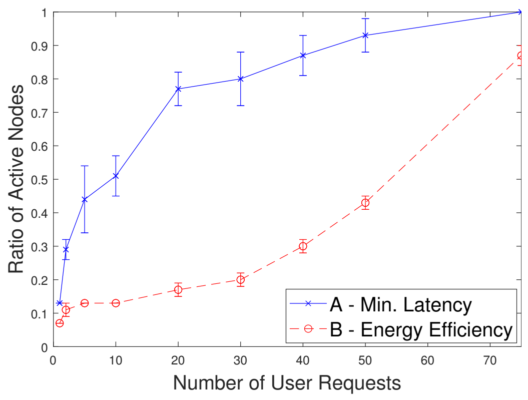


Figure 19: The node utilization rate for the joint use case.

application has been randomized throughout the experiment while increasing user requests. In Fig. 17, the ratio of active gateways for each policy is shown. Higher usage rates are shown for policy *A* since no optimization objective is included regarding gateway efficiency. In contrast, for policy *B*, the ratio of active gateways slightly increases with the increase of connected sensors. Results show that only 51% of the gateways are needed for 750 sensors and that only for 1000 sensors, the ratio is higher than 70%. Policy *A* requires already more than 65% of active gateways for only 50

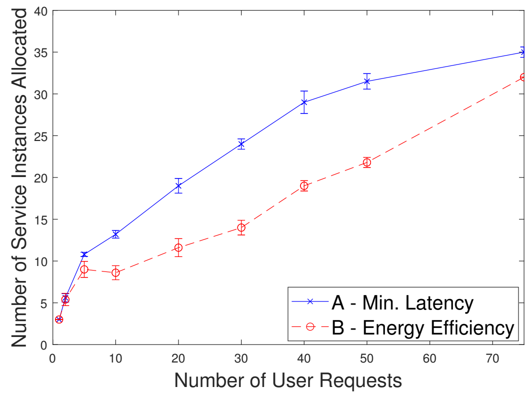


Figure 20: The number of micro-service instances allocated for the joint use case.

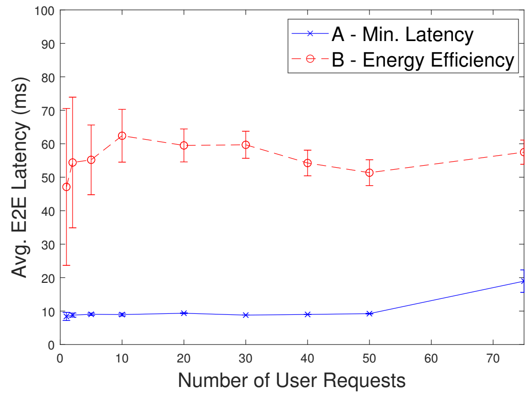


Figure 21: The expected E2E latency for the joint use case.

sensors. Previous use cases showed similar results in terms of gateway usage rates, but in contrast, for this particular scenario, differences are achieved. This occurs because, in the previous use cases, all sensors are assigned to the single application deployed in the network while in this scenario, three applications are deployed and sensors have different application associations. In Fig. 18, the data transfer time per sensor for each policy is illustrated. As expected, the differences between the two strategies have been obtained. For example, for 250 sensors, each sensor requires on average 1.4 seconds to send a message for policy B while for policy A, each sensor takes on average only 130 ms. Policy A can reduce the data transfer time of each sensor at the cost of a higher gateway usage rate. Additionally, in Fig. 19, the ratio of active nodes for each policy is shown. For instance, for 20 user requests, 17% of the nodes are needed to deploy all the required micro-service instances for policy B while for policy A, 77% of nodes are active to fully achieve the optimal solution. For high values of user requests, policy B is not able to reduce the number of active nodes. For example, for 75 user requests, on average 87% of nodes are needed while policy A needs all nodes active in the network to minimize latency. In Fig. 20, the number of micro-service instances allocated in the network for both policies is shown. The differences between both provisioning strategies signifi-

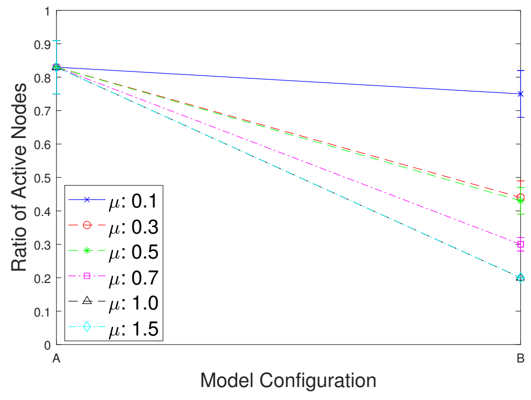


Figure 22: The node utilization rate by changing from policy A to B.

cantly increase for values higher than 5 user requests. In fact, for high values of user requests, policy A deploys 10 more instances than policy B. In Fig. 21, the obtained results in terms of E2E latency are shown. Policy A can minimize the E2E latency since it deploys more micro-service instances, however, for 75 user requests, the infrastructure is exhausted and the latency increases to 19 ms. Policy B by focusing on optimizing energy efficiency is only able to minimize latency to 57 ms for the same number of user requests.

In summary, the Joint Use Case represents a real-world scenario where multiple applications are deployed in the network and cloud providers want to offer their users the best QoS for all of their services. As shown, large differences are achieved in terms of micro-service allocation depending on the provisioning strategy. Cloud providers must determine which requirements are more important for their services to optimize these at the maximum. Clear trade-offs have been presented. Reducing the sensor's data transfer time implies an increase of active gateways and, thus, higher energy costs. Optimizing the energy efficiency in the infrastructure does not translate into low latency values. In fact, to minimize the latency a higher number of service instances needs to be allocated.

6.6. Changing Provisioning Strategies: Impact of Service Migrations

Table 8
The evaluated policies for the dynamic scenario.

	A → B	B → A
Service Migration Factor (μ)	150%	150%
	100%	100%
	70%	70%
	50%	50%
	30%	30%
	10%	10%

The last part of our evaluation considered a use case where cloud providers may want to change their provisioning strategy. Thus, an important factor to measure is the number

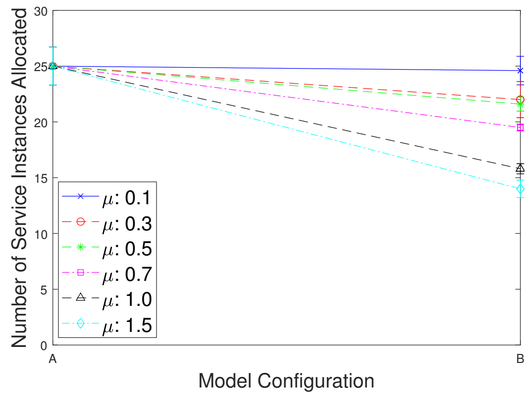


Figure 23: The number of micro-service instances allocated by changing from policy A to B.

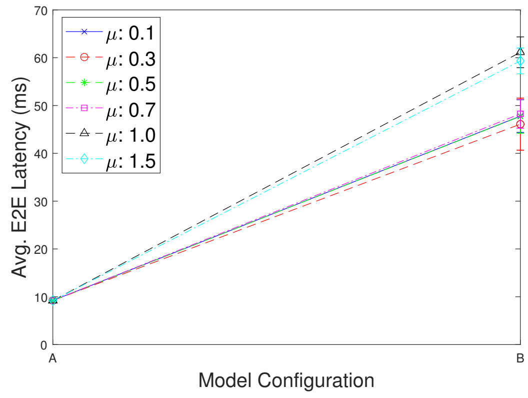


Figure 24: The expected E2E latency per user by changing from policy A to B.

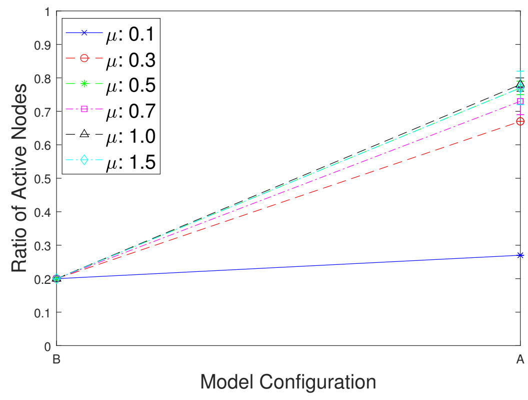


Figure 25: The node utilization rate by changing from policy B to A.

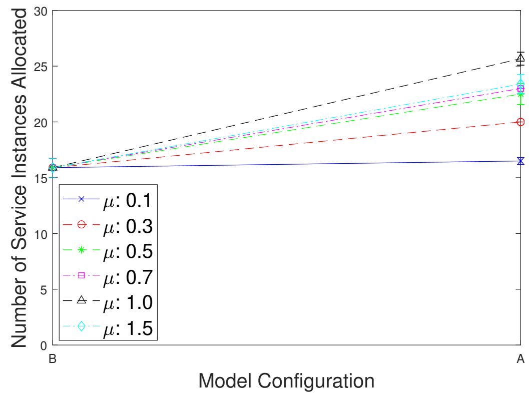


Figure 26: The number of micro-service instances allocated by changing from policy B to A.

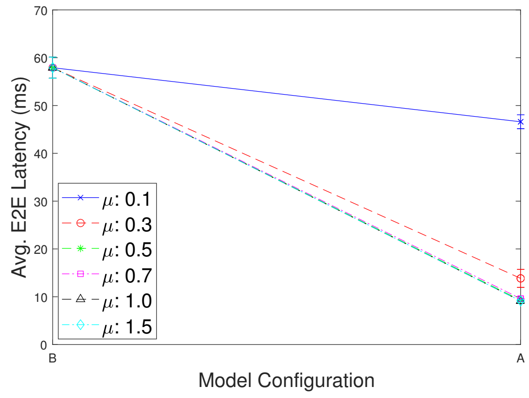


Figure 27: The expected E2E latency per user by changing from policy B to A.

of service migrations needed to find the optimal allocation scheme. For this scenario, the joint use case has been considered with 50 sensors and 30 users. Sensors and user locations have been changed between strategies by attributing new x and y coordinates. Constraints have been included in the model to make sure that the newly calculated positions are inside the evaluation area. The allocation strategy has been changed and different service migration factors have been evaluated as shown in Table 8. It should be highlighted that a migration factor of 150% has been evaluated since in our model a previous iteration may only need to deploy 4 micro-service instances in the network, but a further iteration focused on low latency may allocate 6 micro-service instances. Thus, a migration factor higher than 100% is allowed in our MILP formulation. The impact of service migrations must be studied because it can lead to service disruptions and, thus, it may be desirable to find a sub-optimal solution, in which service migrations are kept to a minimum to reduce the delay caused by service reallocations.

In the evaluation, three factors have been evaluated: the ratio of active nodes, the number of micro-service instances allocated and the expected E2E latency for each user. On the one hand, in Fig. 22, the ratio of nodes by changing from policy A to B is shown while in Fig. 23 and Fig. 24, the number of micro-service instances allocated and the expected E2E latency are shown, respectively. As expected, policy A requires a high number of active nodes to minimize latency and when policy B is applied, it will reallocate micro-service instances to only a small percentage of nodes to maximize energy efficiency. For example, for a service migration factor of 0.5, the number of active nodes is reduced to 43%, which corresponds to a total reduction of 50%, but it increases the expected E2E latency per user from 9ms to 47.7 ms. Furthermore, policy B can significantly reduce the number of micro-service instances allocated in the network from 25 to 14 replicas for the highest migration factor ($\mu = 1.5$). On the other hand, in Fig. 25, the ratio of nodes by changing from policy B to A is shown while in Fig. 26 and Fig. 27, the number of micro-service instances allocated and the expected E2E latency are shown, respectively. As expected, policy B requires a small number of nodes active and when policy A is applied, it will reallocate micro-service instances to reduce latency. For example, even for a small migration factor of 0.3, the expected E2E latency can be reduced from 57 ms to 13.8 ms by deploying on average 4 more micro-service replicas, which increases the node utilization rate from 20% to 66.6%. For higher migration factors, policy A can reduce the expected E2E latency per user between 9 and 10 ms at a cost of a high node utilization rate and a high number of deployed micro-service instances. It should also be noted that for a migration factor of 1.5, the results are similar to the joint use case for the same number of user requests, which was expected.

In summary, this scenario corresponds to a real-world use case where service reallocations may be needed when provisioning strategies are changed. The evaluation has been carried out to quantify what the differences are in terms of service allocation by restricting the MILP model with different migration factors. As shown, significant differences can be achieved. Service providers must decide which allocation strategy is suitable for their services and at what time a different allocation strategy should be applied. Our evaluation proved that ILP-based solutions can provide the optimal solution, however, at the expense of significant execution time. Depending on the problem complexity, several hours might be needed to find the optimal allocation scheme. Nevertheless, service providers can adopt our model as a reference benchmark for their allocation algorithms and they may even calculate the optimal scheme depending on the current network demand. For instance, if they want to change their allocation policy between the two evaluated provisioning strategies, they can use our model to calculate the optimal scheme and then apply it in their network. It might be worth waiting a few hours and properly allocate all resources for this new policy than just quickly change everything and end up with a sub-optimal scheme. Results showed that to change between the two evaluated provisioning strategies a substantial percentage of service migrations are required to find the optimal allocation solution (higher than 70%). Clear trade-offs have been presented between optimizing energy efficiency and minimizing latency. Our model not only allocates service chains, but also deals with sensor's data transfer times and bandwidth requirements which impact the E2E service quality. We believe our model can be executed weekly or even daily depending on the expected demand.

7. Conclusions

In this article, a MILP formulation for the IoT service provisioning problem is presented, which takes SFC concepts, different LPWAN technologies and multiple optimization objectives into account. In recent years, the need for resource provisioning strategies for Fog Computing is increasing due to the deployment of IoT use cases. Billions of connected devices are expected which will make it impractical for current network architectures to support this massive growth since services will be requested on-demand simultaneously by multiple devices at different locations. Cloud providers will need proper service allocation solutions to minimize infrastructure costs and maximize QoS. Our proposed MILP model considers not only cloud requirements but also characteristics stemming from wireless aspects to deal with these demanding requirements. The model considers multiple objectives, such as the acceptance of user requests, user latency reduction or increasing gateway efficiency. To the best of our knowledge, our work goes beyond the current state-of-the-art by providing a complete E2E resource provisioning in Fog-cloud environments while considering both cloud and wireless requirements. Evaluations have been performed to assess the proposed MILP formulation for Smart City use cases. Results show clear trade-offs between the different applied strategies. Cloud providers must decide which allocation strategy is suitable for their services and at what time a different strategy could be applied. Service migrations must also be considered as an important factor in the service allocation. Obtained results show that to change between provisioning strategies a substantial percentage of service migrations are required to find the optimal allocation solution (higher than 70%). The result of our work can serve as a benchmark in research covering IoT provisioning issues in Fog-cloud environments since the model approach is generic, considers several cloud and wireless aspects and can be applied to a wide range of IoT use cases. Future heuristics can be evaluated based on the measured execution times (e.g. latency-aware algorithms vs the *MIN UL* objective). As future work, the MILP model will be validated through real service deployments. Additionally, dynamic user demands will be studied, which will allow our approach to learn from network behavior.

Acknowledgment

This research was performed within the project "Intelligent DEnse And Longe range IoT networks (IDEAL-IoT)" under Grant Agreement #S004017N, from the fund for Scientific Research-Flanders (FWO-V). The computational resources (Stevin Supercomputer Infrastructure) and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by Ghent University, FWO and the Flemish Government – department EWI.

References

[1] M. Chiang, T. Zhang, Fog and iot: An overview of research opportunities, IEEE Internet of Things Journal 3 (2016) 854–864.

[2] B. N. Silva, M. Khan, K. Han, Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities, Sustainable Cities and Society 38 (2018) 697–713.

[3] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022 White Paper, 2019.

[4] M. Mukherjee, L. Shu, D. Wang, Survey of fog computing: Fundamental, network applications, and research challenges, IEEE Communications Surveys & Tutorials 20 (2018) 1826–1857.

[5] P. Hu, S. Dhelim, H. Ning, T. Qiu, Survey on fog computing: architecture, key technologies, applications and open issues, Journal of network and computer applications 98 (2017) 27–42.

[6] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, A. V. Vasilakos, Fog computing for sustainable smart cities: A survey, ACM Computing Surveys (CSUR) 50 (2017) 32.

[7] K. Mekki, E. Bajic, F. Chaxel, F. Meyer, A comparative study of lpwan technologies for large-scale iot deployment, ICT express 5 (2019) 1–7.

[8] A. J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, N. Strachan, Evaluation of lora and lorawan for wireless sensor networks, in: 2016 IEEE SENSORS, IEEE, pp. 1–3.

[9] J. C. Zuniga, B. Ponsard, Sigfox system description, LPWAN@ IETF97, Nov. 14th 25 (2016).

[10] M. Park, Ieee 802.11 ah: sub-1-ghz license-exempt operation for the internet of things, IEEE Communications Magazine 53 (2015) 145–151.

[11] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, L. Safina, Microservices: yesterday, today, and tomorrow, in: Present and ulterior software engineering, Springer, 2017, pp. 195–216.

[12] D. Bhamare, R. Jain, M. Samaka, A. Erbad, A survey on service function chaining, Journal of Network and Computer Applications 75 (2016) 138–155.

[13] Y. Xie, Z. Liu, S. Wang, Y. Wang, Service function chaining resource allocation: A survey, arXiv preprint arXiv:1608.00095 (2016).

[14] J. Santos, T. Wauters, B. Volckaert, F. De Turck, Towards delay-aware container-based service function chaining in fog computing, in: NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, IEEE, pp. 1–9.

[15] T. Verbelen, P. Simoens, F. De Turck, B. Dhoedt, Leveraging cloudlets for immersive collaborative applications, IEEE Pervasive Computing 12 (2013) 30–38.

[16] A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future generation computer systems 28 (2012) 755–768.

[17] L. Deboosere, B. Vankeirsbilck, P. Simoens, F. De Turck, B. Dhoedt, P. Demeester, Efficient resource management for virtual desktop cloud computing, The Journal of Supercomputing 62 (2012) 741–767.

[18] A. Brogi, S. Forti, Qos-aware deployment of iot applications through the fog, IEEE Internet of Things Journal 4 (2017) 1185–1192.

[19] O. Skarlat, S. Schulte, M. Borkowski, P. Leitner, Resource provisioning for iot services in the fog, in: 2016 IEEE 9th international conference on service-oriented computing and applications (SOCA), IEEE, pp. 32–39.

[20] O. Skarlat, M. Nardelli, S. Schulte, S. Dustdar, Towards qos-aware fog service placement, in: 2017 IEEE 1st international conference on Fog and Edge Computing (ICFEC), IEEE, pp. 89–96.

[21] S. Agarwal, S. Yadav, A. K. Yadav, An efficient architecture and algorithm for resource provisioning in fog computing, International Journal of Information Engineering and Electronic Business 8 (2016) 48.

[22] A. Yasmeen, N. Javaid, O. U. Rehman, H. Iftikhar, M. F. Malik, F. J. Muhammad, Efficient resource provisioning for smart buildings utilizing fog and cloud based environment, in: 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC), IEEE, pp. 811–816.

[23] J. Yao, N. Ansari, Qos-aware fog resource provisioning and mobile device power control in iot networks, IEEE Transactions on Network and Service Management 16 (2018) 167–175.

[24] J. Yao, N. Ansari, Fog resource provisioning in reliability-aware iot networks, IEEE Internet of Things Journal (2019).

[25] N. Kherraf, H. A. Alameddine, S. Sharafeddine, C. Assi, A. Ghrayeb, Optimized provisioning of edge computing resources with heterogeneous workload in iot networks, IEEE Transactions on Network and Service Management (2019).

[26] F. Chiti, R. Fantacci, F. Paganelli, B. Picano, Virtual functions placement with time constraints in fog computing: a matching theory perspective, IEEE Transactions on Network and Service Management (2019).

[27] H. R. Arkian, A. Diyanat, A. Pourkhalili, Mist: Fog-based data analytics scheme with cost-efficient resource provisioning for iot crowdsensing applications, Journal of Network and Computer Applications 82 (2017) 152–165.

[28] X. Chen, S. Leng, K. Zhang, K. Xiong, A machine-learning based time constrained resource allocation scheme for vehicular fog computing, China Communications 16 (2019) 29–41.

[29] M. Etemadi, M. Ghobaei-Arani, A. Shahidinejad, Resource provisioning for iot services in the fog computing environment: An autonomic approach, Computer Communications (2020).

[30] R. K. Naha, S. Garg, A. Chan, S. K. Battula, Deadline-based dynamic resource allocation and provisioning algorithms in fog-cloud environment, Future Generation Computer Systems 104 (2020) 131–141.

[31] J. Santos, T. Wauters, B. Volckaert, F. De Turck, Resource provisioning for iot application services in smart cities, in: 2017 13th International Conference on Network and Service Management (CNSM), IEEE, pp. 1–9.

[32] S. Dawaliby, A. Bradai, Y. Pousset, R. Riggio, Dynamic network slicing for lorawan, in: 2018 14th International Conference on Network and Service Management (CNSM), IEEE, pp. 134–142.

[33] T. Adame, A. Bel, B. Bellalta, J. Barcelo, M. Oliver, Ieee 802.11 ah: the wifi approach for m2m communications, IEEE Wireless Communications 21 (2014) 144–152.

[34] E. Khorov, A. Lyakhov, A. Krotov, A. Guschin, A survey on ieee 802.11 ah: An enabling networking technology for smart cities, Computer Communications 58 (2015) 53–69.

[35] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, T. Watteyne, Understanding the limits of lorawan, IEEE Communications magazine 55 (2017) 34–40.

[36] S. Aust, T. Ito, Sub 1ghz wireless lan propagation path loss models for urban smart grid applications, in: 2012 International Conference on Computing, Networking and Communications (ICNC), IEEE, pp. 116–120.

[37] P. Jörke, S. Böcker, F. Liedmann, C. Wietfeld, Urban channel models for smart city iot-networks based on empirical measurements of lora-links at 433 and 868 mhz, in: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), IEEE, pp. 1–6.

[38] A. Šljivo, D. Kerkhove, L. Tian, J. Famaey, A. Munteanu, I. Moerman, J. Hoebeke, E. De Poorter, Performance evaluation of ieee 802.11 ah networks with high-throughput bidirectional traffic, Sensors 18 (2018) 325.

[39] J. Santos, T. Vanhove, M. Sebrechts, T. Dupont, W. Kerckhove, B. Braem, G. Van Seghbroeck, T. Wauters, P. Leroux, S. Latre, et al., City of things: enabling resource provisioning in smart cities, IEEE Communications Magazine 56 (2018) 177–183.

[40] A. Medvedev, P. Fedchenkov, A. Zaslavsky, T. Anagnostopoulos, S. Khoruzhnikov, Waste management as an iot-enabled service in smart cities, in: Internet of Things, Smart Spaces, and Next Generation Networks and Systems, Springer, 2015, pp. 104–115.

[41] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, M. Satyanarayanan, A scalable and privacy-aware iot service for live video analytics, in: Proceedings of the 8th ACM on Multimedia Systems Conference, ACM, pp. 38–49.

[42] J. Santos, P. Leroux, T. Wauters, B. Volckaert, F. De Turck, Anomaly detection for smart city applications over 5g low power wide area networks, in: NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium, IEEE, pp. 1–9.

[43] Obelisk, a platform for building scalable applications on iot centric timeseries data, 2020.

[44] Y. Cheng, X. Li, Z. Li, S. Jiang, Y. Li, J. Jia, X. Jiang, Aircloud: a cloud-based air-quality monitoring system for everyone, in: Pro-

ceedings of the 12th ACM Conference on Embedded Network Sensor Systems, ACM, pp. 251–265.
[45] I. ILOG, Ibm cplex ilog optimization studio, 2020.
[46] HPC-UGent, High performance computing infrastructure, 2020.

José Santos obtained his M.Sc. degree in Electrical and Computers Engineering in July 2015 from University of Porto, Portugal. He is currently a PhD Student of Computer Science in the Internet Technology and Data Science Lab (IDLab) Research Group at Ghent University - imec, Belgium. Before joining IDLab, he was a Research Intern at PROEF Group where he was involved in EU-funded projects. His research interests include Cloud Computing, IoT and Software-Defined Networking.

Tim Wauters obtained his M.Sc. and Ph.D. degrees in electro-technical engineering from Ghent University in 2001 and 2007 respectively. He has been working as a post-doctoral fellow in the Department of Information Technology (INTEC) at Ghent University, and is now also active as a senior researcher at imec. His main research interests focus on network and service architectures for multimedia delivery services. His work has been published in about 80 scientific publications.

Bruno Volckaert is a professor in software engineering in the Department of Information Technology (INTEC) at Ghent University and senior researcher at imec. He obtained his PhD at Ghent University on data intensive scheduling and service management for Grid computing in 2006. He has worked on over 45 national and international research projects and is author or co-author of more than 130 papers published in international journals and conference proceedings.

Filip De Turck leads the network and service management research group at the Department of Information Technology of the Ghent University, Belgium and imec. He (co-) authored over 450 peer-reviewed papers and his research interests include the design of efficient virtualized network and cloud systems. He serves as Chair of the IEEE Technical Committee on Network Operations and Management (CNOM), and is on the TPC of many network and service management conferences.